R Programming Language
R04 Graphics

林 建 甫
C.F. Jeff Lin, MD. PhD.

台北大學統計系助理教授
台北榮民總醫院生物統計顧問
美國密西根大學生物統計博士

Graphics in R

Graphics in R

- High level:
  - plot(), boxplot(), hist(), dotchart(), pie(),
  - qqplot(), pairs(), coplot
- Low level
  - points(), lines(), text(), abline(), legend(), title(), polygon(), axis(), cex()
- Interactive graphics
  - locator(), identify()

Graphics Elements par()

- axes=TRUE, type="p", pch=1", lty="1"
- xlim=, ylim=
- xlab=, ylab=
- main=
- sub=

Example Data

- A data set contains 80 body weights from a sample of children measured as closely as possible to their fifth birthday and serves to illustrated a few fundamental descriptive statistical summaries.
- Read in data

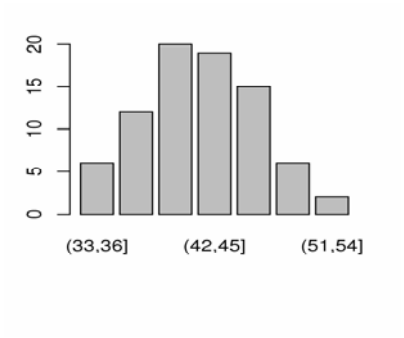
```
BWdata<-read.csv("selvinB0101csv.csv",
        header = TRUE, sep = ",", dec=".")
```


33.1	33.5	34.7	35.2	35.5	35.7
36.1	36.5	37.0	37.0	37.2	37.5
38.2	38.5	38.7	38.7	38.8	38.9
39.2	39.2	39.2	39.5	39.5	39.7
39.7	39.7	40.0	40.0	40.2	40.7
40.7	41.0	41.0	41.0	41.2	41.2
41.2	41.7	42.1	42.1	42.2	42.2
42.5	42.7	43.0	43.0	43.2	43.5
43.5	43.5	43.5	44.0	44.0	44.2
44.5	44.8	44.9	45.1	45.2	45.3
45.3	45.5	46.0	46.0	46.5	46.7
47.0	47.0	47.0	47.2	47.6	47.8
48.1	48.2	49.0	50.0	50.2	50.5
51.6	52.7				

Bar Chart: barplot()

```
> # Bar plot
> BWdata<-read.csv("selvinB0101csv.csv",
  header = TRUE, sep = ",", dec=".")
> attach(BWdata)
> BWpound
> BWcut<-cut(BWpound, breaks=seq(33,54,by=3))
> BWcut
> BWcutTable<-table(BWcut)
> BWcutTable
> barplot(BWcutTable)
```

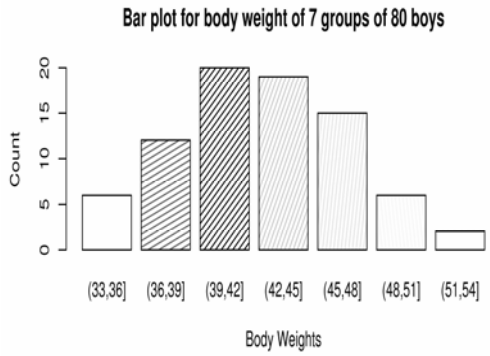

Bar Chart



Put Some Colors

```
> # Put some shaddings and lines
  > barplot(BWcutTable, density=20,
    angle=seq(0,by=20,length=length(BWcutTable)),
    xlab="Body Weights",ylab="Count",
    main="Barplot for body weight of 7 groups of 80 boys")
  > # Put some colors
  > colorname<-colors()
  > colorname
  [1] "white"              "aliceblue"          "antiquewhite"
  [4] "antiquewhite1"      "antiquewhite2"      "antiquewhite3"
  > barplot(BWcutTable,  xlab="Body Weights",
    ylab="Count", main="Barplot for body weight of 7 groups of
    80 boys",
    col = colorname[seq(1,by=130,length=length(BWcutTable))],
    density=20, angle=seq(0,by=20,length=length(BWcutTable)))
```

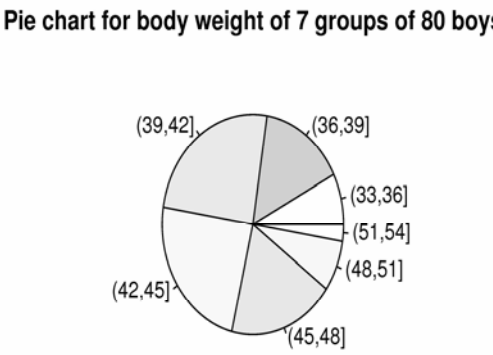

Put Some Colors



Pie Graphics: pie()

```
> BWdata<-read.csv("selvinB0101csv.csv",
    header = TRUE, sep = ",", dec=".")
  > attach(BWdata)
  > BWpound
  > BWcut<-cut(BWpound, breaks=seq(33,54,by=3))
  > BWcut
  > BWcutTable<-table(BWcut)
  > BWcutTable

> pie(BWcutTable, main="Pie chart for body weight of 7
    groups of 80 boys")
```


Pie Chart



Pareto Plot

> BWcutTableSort<-sort(BWcutTable)

> BWcutTableSort<-BWcutTableSort[7:1]

> BWcutTableSort

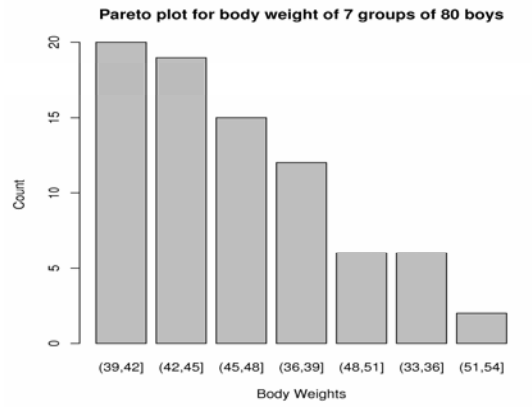
BWcut

(39,42] (42,45] (45,48] (36,39] (48,51] (33,36] (51,54]

20 19 15 12 6 6 2

> barplot(BWcutTableSort, xlab="Body Weights", ylab="Count", main="Pareto plot for body weight of 7 groups of 80 boys")

Pareto Plot



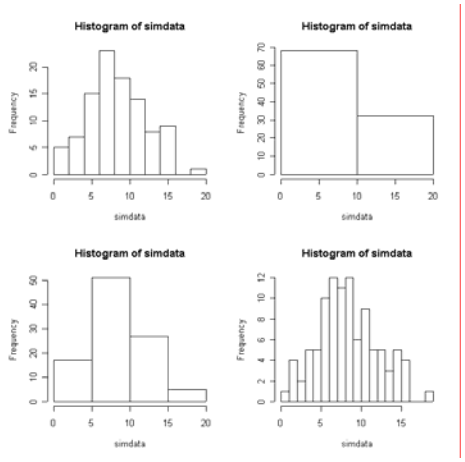
Histogram

- A *histogram* is a special kind of bar plot
- It allows you to visualize the *distribution* of values for a numerical variable
- When drawn with a *density scale:*
  - the *AREA* (NOT height) of each bar is the proportion of observations in the interval
  - the *TOTAL AREA* is 100% (or 1)

R: Making a Histogram

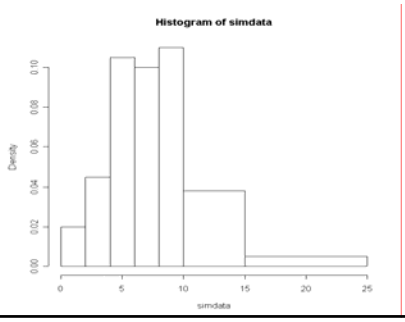
- Type **`?hist`** to view the help file
  - Note some important arguments, esp **`breaks`**
- Simulate some data, make histograms varying the number of bars (also called 'bins' or 'cells'), e.g.

```
> par(mfrow=c(2,2))   # set up
  multiple plots
> simdata <-rchisq(100,8)
> hist(simdata)  # default number of
  bins
> hist(simdata,breaks=2)  # etc,4,20
```

R: setting your own breakpoints

```
> bps <- c(0,2,4,6,8,10,15,25)
> hist(simdata,breaks=bps)
```



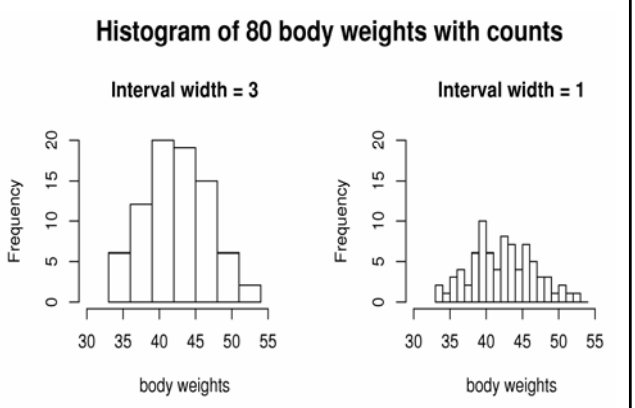
Histogram: hist()

```
> hist(BWpound,freq=FALSE)
  > hist(BWpound,freq=FALSE, main="Histogram of 80 body
    weights with relative frequency")

  > hist(BWpound,main="Histogram of 80 body weights with
    counts")
  > hist(BWpound,freq=FALSE)
  > hist(BWpound,breaks=seq(33,54,1), freq=FALSE,
    main="Histogram of 80 body weights with relative frequency")

  > hist(BWpound,break=seq(33,54,1),freq=FALSE)
  > hist(BWpound,nclass=25)
  > hist(BWpound,nclass=50)
```


Histogram: hist() Different Interval Widths

```
> # Count as Frequency
> par(mfrow=c(1,2),omi=c(0,0,1,0))

> hist(BWpound,breaks=seq(33,54,3), main="Interval
  width = 3", xlab="body weights",xlim=c(30,57),
  ylim=c(0,0.15))
>
> hist(BWpound,breaks=seq(33,54,1), main="Interval
  width = 1", xlab="body weights",xlim=c(30,57),
  ylim=c(0,0.15))
>
> mtext("Histogram of 80 body weights with counts",
   side=3,outer=TRUE, font=2,cex=1.5)
```

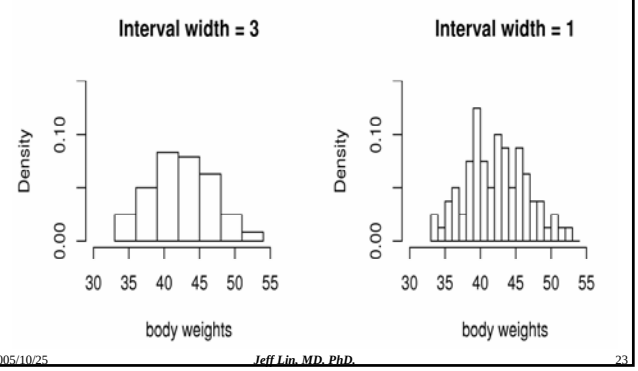

Histogram: Different Interval Widths



Histogram: Relative Frequency

```
> # Relative Frequency
  > par(mfrow=c(1,2),omi=c(0,0,1,0))
  > hist(BWpound,breaks=seq(33,54,3), freq=FALSE,
    main="Interval width = 3",
     xlab="body weights",xlim=c(30,57), ylim=c(0,0.15))
  >
  > hist(BWpound,breaks=seq(33,54,1), freq=FALSE,
    main="Interval width = 1",
     xlab="body weights",xlim=c(30,57), ylim=c(0,0.15))
  >
  > mtext("Histogram of 80 body weights with relative
    frequency",
     side=3,outer=TRUE, font=2,cex=1.5)
```

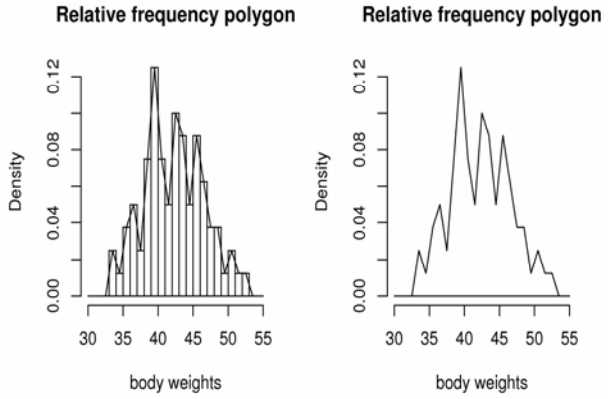

Histogram: Relative Frequency



Relative Frequency Polygon

```
> par(mfrow=c(1,2))
> BWhist<-
  hist(BWpound,breaks=seq(30,55,1),freq=FALSE,
   main="Relative frequency polygon",xlab="body
  weights")
> lines(BWhist$mid,BWhist$intensities)
> abline(h=0)
>
> BWhist<-
  hist(BWpound,breaks=seq(30,55,1),freq=FALSE,
  border="white",
   main="Relative frequency polygon",xlab="body
  weights")
> lines(BWhist$mid,BWhist$intensities)
> abline(h=0)
```

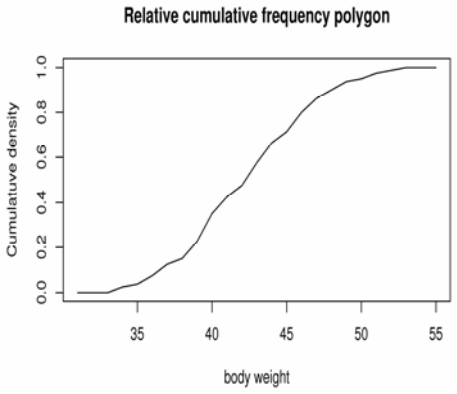
Relative Frequency Polygon



Relative Cumulative Frequency Polygon

```
> par(mfrow=c(1,1))
> BWhist<-
  hist(BWpound,breaks=seq(30,55,1),freq=FALSE,plot=
  FALSE)
> BWpoundRCF<-cumsum(BWhist$intensities)
> BWpoundRCF
> BWpoundRCF
 [1] 0.0000 0.0000 0.0000 0.0250 0.0375 0.0750 0.1250 0.1500 0.2250 0.3500
[11] 0.4250 0.4750 0.5750 0.6625 0.7125 0.8000 0.8625 0.9000 0.9375 0.9500
[21] 0.9750 0.9875 1.0000 1.0000 1.0000
> BWint<-seq(31,55,1)
> plot(BWint,BWpoundRCF, main="Relative cumulative
  frequency polygon",
   xlab="body weight",type="l",ylab="Cumulative
  density")
```

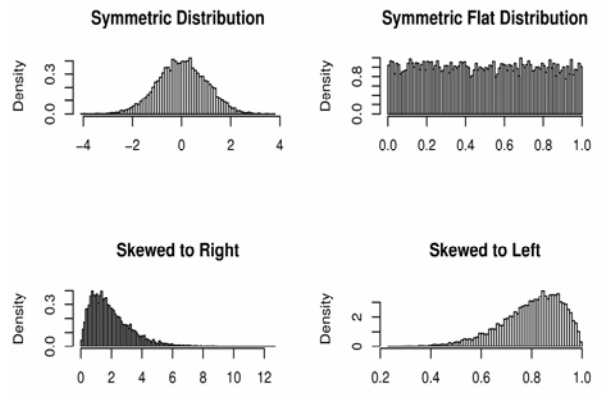

Relative Cumulative Frequency Polygon



Histrogram and Shape of Distribution

```
> par(mfrow=c(2,2))
> histSYM<-hist(rnorm(10000),nclas=100,freq=FALSE,
   main="Symmetric Distribution", xlab="")
> histFLAT<-hist(runif(10000),nclas=100,freq=FALSE,
   main="Symmetric Flat Distribution", xlab="")
> histSKR<-
  hist(rgamma(10000,shape=2,scale=1),freq=FALSE,
  nclas=100,
   main="Skewed to Right", xlab="")
> histSKL<-hist(rbeta(10000,8,2),nclas=100,freq=FALSE,
   main="Skewed to Left", xlab="")
```


Histrogram and Shape of Distribution



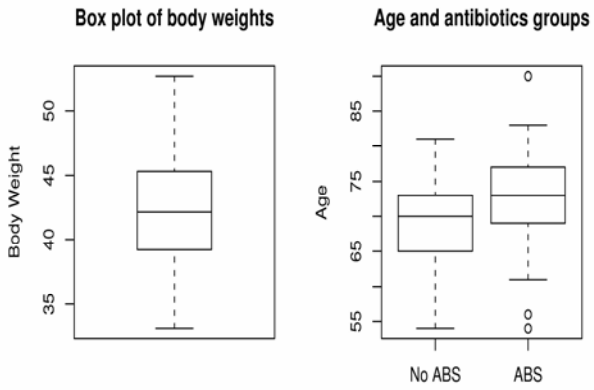
Stem-and-Leaf Plot: stem()

```
> # Stem-and-leaf plot of 80 five-year-old boy body weights
> stem(BWpound)
The decimal point is at the |
  32 | 15
  34 | 7257
  36 | 150025
  38 | 25778922255777
  40 | 002770002227
  42 | 1122570025555
  44 | 00258912335
  46 | 0057000268
  48 | 120
  50 | 0256
  52 | 7
```

Box Plot: boxplot()

```
> par(mfrow=c(1,2))
> boxplot(BWpound, ylab="Body
  Weight",main="Box plot of body weights")
>
> DMTKRdata<-read.csv("DMTKRcsv.csv",
  header = TRUE, sep = ",", dec=".")
> attach(DMTKRdata)
> boxplot(age~ABS,names=c("No
  ABS","ABS"), ylab="Age",
  main="Age and antibiotics groups")
```


Box Plot: boxplot()



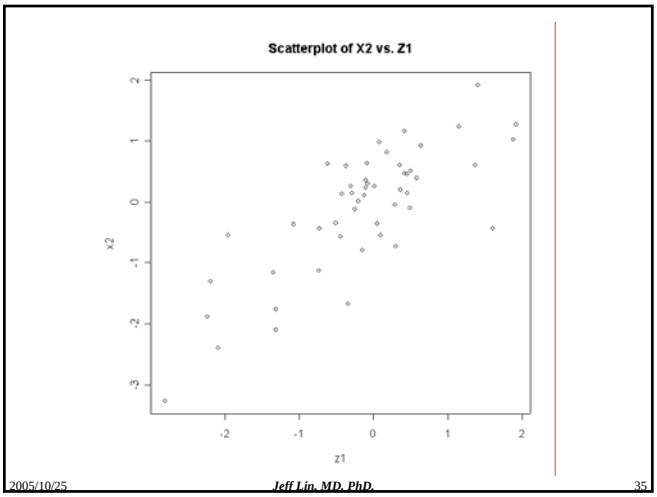
Scatterplot

- A scatterplot is a standard two-dimensional (X,Y) plot
- Used to examine the relationship between two (continuous) variables
- It is often useful to plot values for a single variable against the order or time the values were obtained

R: Making a Scatterplot

- Type **`?plot`** to view the help file
  - For now we will focus on simple plots, but R allows extensive user control for highly customized plots
- Simulate a bivariate data set:

```
> z1 <- rnorm(50)
> z2 <- rnorm(50)
> rho <- .75          # (or any number
   between –1 and 1)
> x2<- rho*z1+sqrt(1-rho^2)*z2
> plot(z1,x2)
```

Plot X and Y: plot()

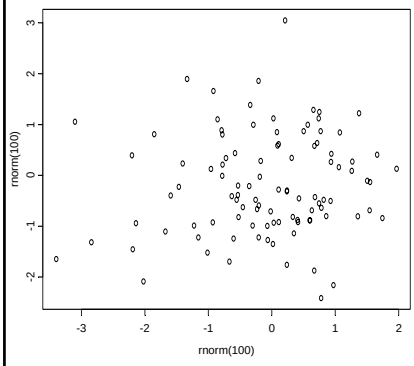
- If x and y are vectors, plot(x,y) produces a scatterplot of x against y.
- plot(x) produces a time series plot if x is a numeric vector or time series object.

plot()

- plot(df), plot(~ expr), plot(y ~ expr), where df is a data frame, y is any object, expr is a list of object names separated by `+' (e.g. a + b + c).
- The first two forms produce distributional plots of the variables in a data frame (first form) or of a number of named objects (second form). The third form plots y against every object named in expr.

Graphics with plot()

```
> plot(rnorm(100),rnorm(100))
```



The function **`rnorm()`** Simulates a random normal distribution .

Help **`?rnorm`**, and **`?runif`**, **`?rexp`**, **`?binom`**, ...

Plot Options

- **par(mfrow=c(2,2))**
  - **plot 2x2 plots**
- **ylim, xlim**
  - **decide x and y axis ranges**
- **xlab="x", ylab="Sin x"**
  - **put label on axes**
- **main="A Line"**
  - **put title**

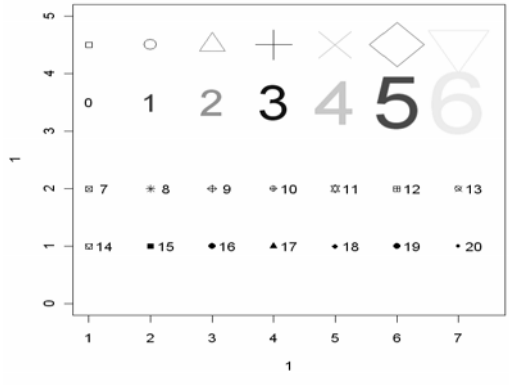
Multiple Plots

- **par(mfrow=c(2,2), pch=16)**

Adding points, lines, and text Size, Colour and Choice of Plotting Symbol

```
> plot(1, 1, xlim=c(1, 7.5), ylim=c(0,5), type="n")
# Do not plot points
> points(1:7, rep(4.5, 7), cex=1:7, col=1:7, pch=0:6)
> text(1:7,rep(3.5, 7), labels=paste(0:6), cex=1:7, col=1:7)
> points(1:7,rep(2,7), pch=(0:6)+7)
# Plot symbols 7 to 13
> text((1:7)+0.25, rep(2,7),  paste((0:6)+7))
# Label with symbol number
> points(1:7,rep(1,7), pch=(0:6)+14) # Plot symbols 14 to
    20
> text((1:7)+0.25, rep(1,7), paste((0:6)+14))
# Labels with symbol number
```


Adding points, lines, and text Size, Colour and Choice of Plotting Symbol



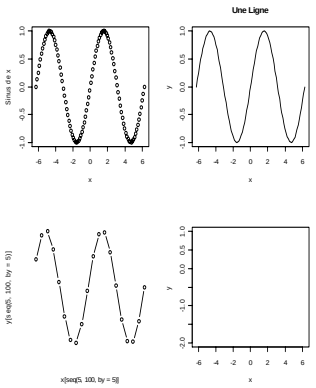
Colors

```
> # A function to demonstrate plot options
> view.colours <- function(){
    plot(1, 1, xlim=c(0,14), ylim=c(0,3), type="n", axes=F,
    xlab="",ylab="")
    text(1:6, rep(2.5,6), paste(1:6), col=palette()[1:6], cex=2.5)
    text(10, 2.5, "Default palette", adj=0)
    rainchars <- c("R","O","Y","G","B","I","V")
    text(1:7, rep(1.5,7), rainchars, col=rainbow(7), cex=2.5)
    text(10, 1.5, "rainbow(7)", adj=0)
    x cmtxt <- substring("cm.colors", 1:9,1:9)
    # Split "cm.colors" into its 9 characters
    text(1:9, rep(0.5,9), cmtxt, col=cm.colors(9), cex=3)
    text(10, 0.5, "cm.colors(9)", adj=0)
    }
> view.colours()
```


Colors

Graphics with `plot()`

```
> x <- seq(-2*pi,2*pi,length=100)
> y <- sin(x)

> par(mfrow=c(2,2))
> plot(x,y,xlab="x",
    ylab="Sin x")

> plot(x,y,type= "l",
    main="A Line")

> plot(x[seq(5,100,by=5)],
    y[seq(5,100,by=5)],
    type= "b",axes=F)

> plot(x,y,type="n",
    ylim=c(-2,1)
> par(mfrow=c(1,1))
```



Graphical Parameters of plot()

type = “c”: c =p (default), l, b,s,o,h,n.
pch=“+” : character or numbers 1 – 18
lty=1 : numbers
lwd=2 : numbers
axes = “L”: L= F, T
xlab =“string”, ylab=“string”
sub = “string”, main =“string”
xlim = c(lo,hi), ylim= c(lo,hi)
And some more.

Other Graphical Functions

`> axis(1,at=c(2,4,5), legend("A","B","C"))`	Axis details (“ticks”, légende, …) Use `xaxt="n"` ou `yaxt="n"` inside `plot()`
`> lines(x,y,…)`	Line plots
`> abline(lsfit(x,y))` `> abline(0,1)`	Add an adjustment add a line of slope 1 and intercept 0
`> legend(locator(1),…)`	Legends: very flexible

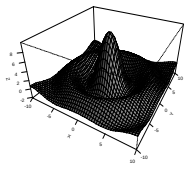
Graphical Parameters of plot()

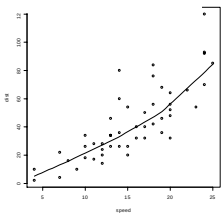
```
x <- 1:10
y <- 2*x + rnorm(10,0,1)
plot(x,y,type="p") #Try l,b,s,o,h,n
   # axes=T, F
   #  xlab="age", ylab="weight"
   # sub="sub title", main="main title"
   # xlim=c(0,12), ylim=c(-1,12)
```

Other Graphical Functions

See also:

```
barplot()
image()
hist()
pairs()
persp()
piechart()
polygon()

library(modreg)
scatter.smooth()
```





Plots for Multivariate Data

```
pairs(stack.x)
x <- 1:20/20
y <- 1:20/20
z <-
   outer(x,y,function(a,b){cos(10*a*b)/(1+a*b^
   2)})
contour(x,y,z)
persp(x,y,z)
image(x,y,z)
```


Interactive Graphics Functions

- locator(n,type="p") :Waits for the user to select locations on the current plot using the left mouse button. This continues until n (default 500) points have been selected.
- identify(x, y, labels) : Allow the user to highlight any of the points defined by x and y.
- text(x,y,"Hey"): Write text at coordinate x,y.

Frequently used operators

`<-`	Assign
`+`	Sum
`-`	Difference
`*`	Multiplication
`/`	Division
`^`	Exponent
`%%`	Mod
`%*%`	Dot product
`%/%`	Integer division
`%in%`	Subset

`|`	Or
`&`	And
`<`	Less
`>`	Greater
`<=`	Less or =
`>=`	Greater or =
`!`	Not
`!=`	Not equal
`==`	Is equal

Frequently used functions

`c`	Concatenate
`cbind, rbind`	Concatenate vectors
`min`	Minimum
`max`	Maximum
`length`	# values
`dim`	# rows, cols
`floor`	Max integer in
`which`	TRUE indices
`table`	Counts

`summary`	Generic stats
`Sort, order, rank`	Sort, order, rank a vector
`print`	Show value
`cat`	Print as char
`paste`	`c()` as char
`round`	Round
`apply`	Repeat over rows, cols

Statistical functions

`rnorm, dnorm, pnorm, qnorm`	Normal distribution random sample, density, cdf and quantiles
`lm, glm, anova`	Model fitting
`loess, lowess`	Smooth curve fitting
`sample`	Resampling (bootstrap, permutation)
`.Random.seed`	Random number generation
`mean, median`	Location statistics
`var, cor, cov, mad, range`	Scale statistics
`svd, qr, chol, eigen`	Linear algebra

Graphical functions

plot	Generic plot eg: scatter
points	Add points
lines, abline	Add lines
text, mtext	Add text
legend	Add a legend
axis	Add axes
box	Add box around all axes
par	Plotting parameters (lots!)
colors, palette	Use colors

Lattice Graphics

- # S-plus trellis library
- # R: load lattice package

Thanks !