数据分析与处理技术

数据可视化：基础图形系统

图形设备

数据可视化首先要解决一个问题：图形创建在哪里?

原理：只要在R中创建图形，就会返回到一个设备上。设个设备包括：PDF、PNG、JPEG或位图。

```
> pdf('mytestgraph.pdf')
> plot(cars)
> dev.off()
RStudioGD
        2
```

这是使用pdf()函数打开一个pdf文件，将图形输出到pdf文件里，做图结束后关闭文件。

同样也存在输出到其他类型图片文件中的函数如：png() jpeg( )

打开新图形窗口设备：

```
> dev.new()
```

每一个图形设备都有编号，当前只有一个设备处于活动状态，如下设置4号设备为活动状态

```
> dev.set(4)
```

查看当前哪个设备是活动设备

```
> dev.cur()
```

图形设备打开时，R为我们配备了一套齐全的默认参数设置，即**图形元素**已经备齐，例如以前做图中用到的颜色、线形等。

```
> par()      #查看默认图形参数属性

> opar<-par(no.readonly = T)    #图形参数可以读出并保留进变量中，以备恢复

> par(opar)          #par将括号的参数设置设为当前设备的默认参数
```

数据可视化中的图形元素

数据可视化不同于艺术绘画，它必须

- 满足数据的精确表达要求
- 绘图直观易懂

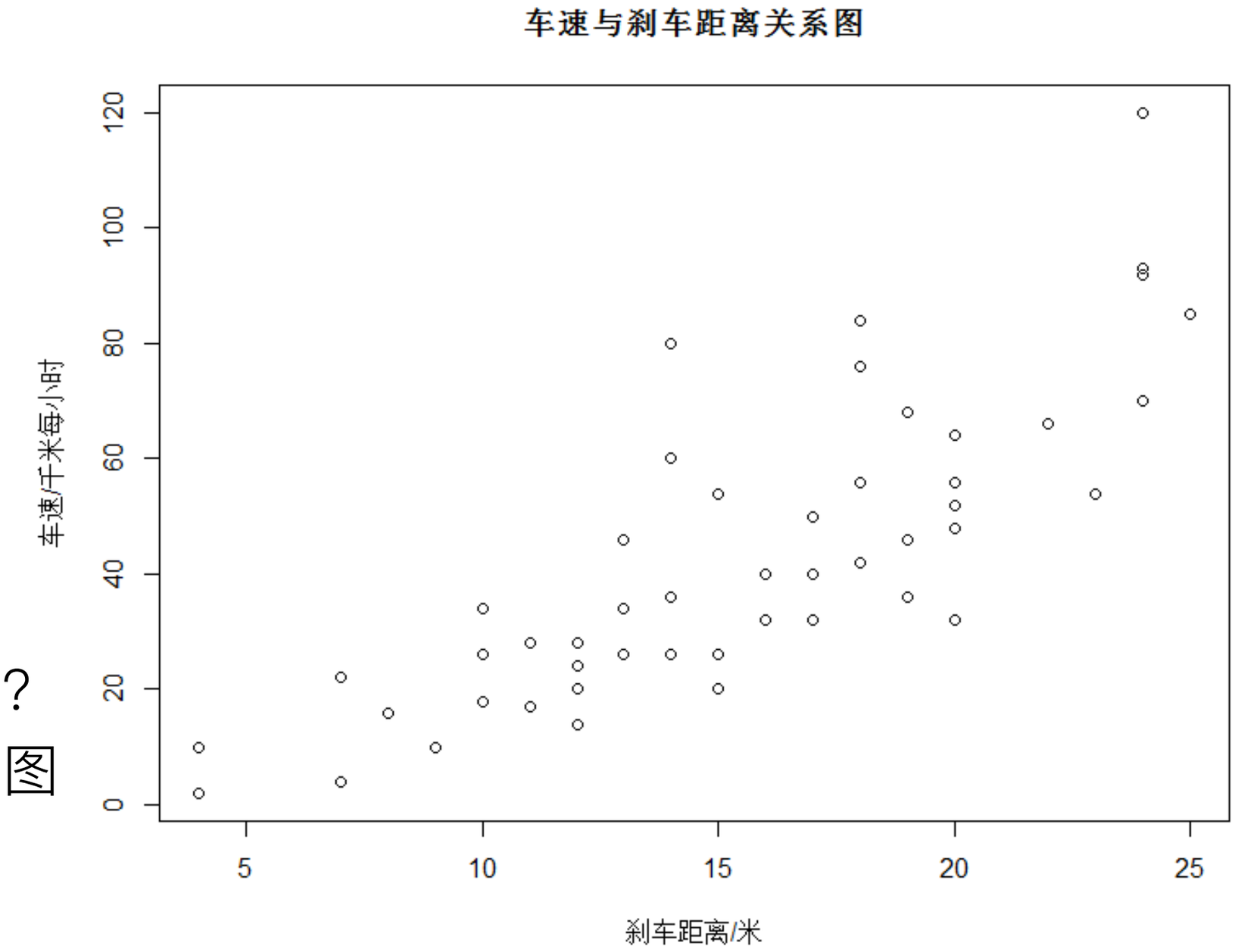
调用准备好的图形元素组合成需要的图形

问题：图形画在哪里?

问题：图形元素都有什么?

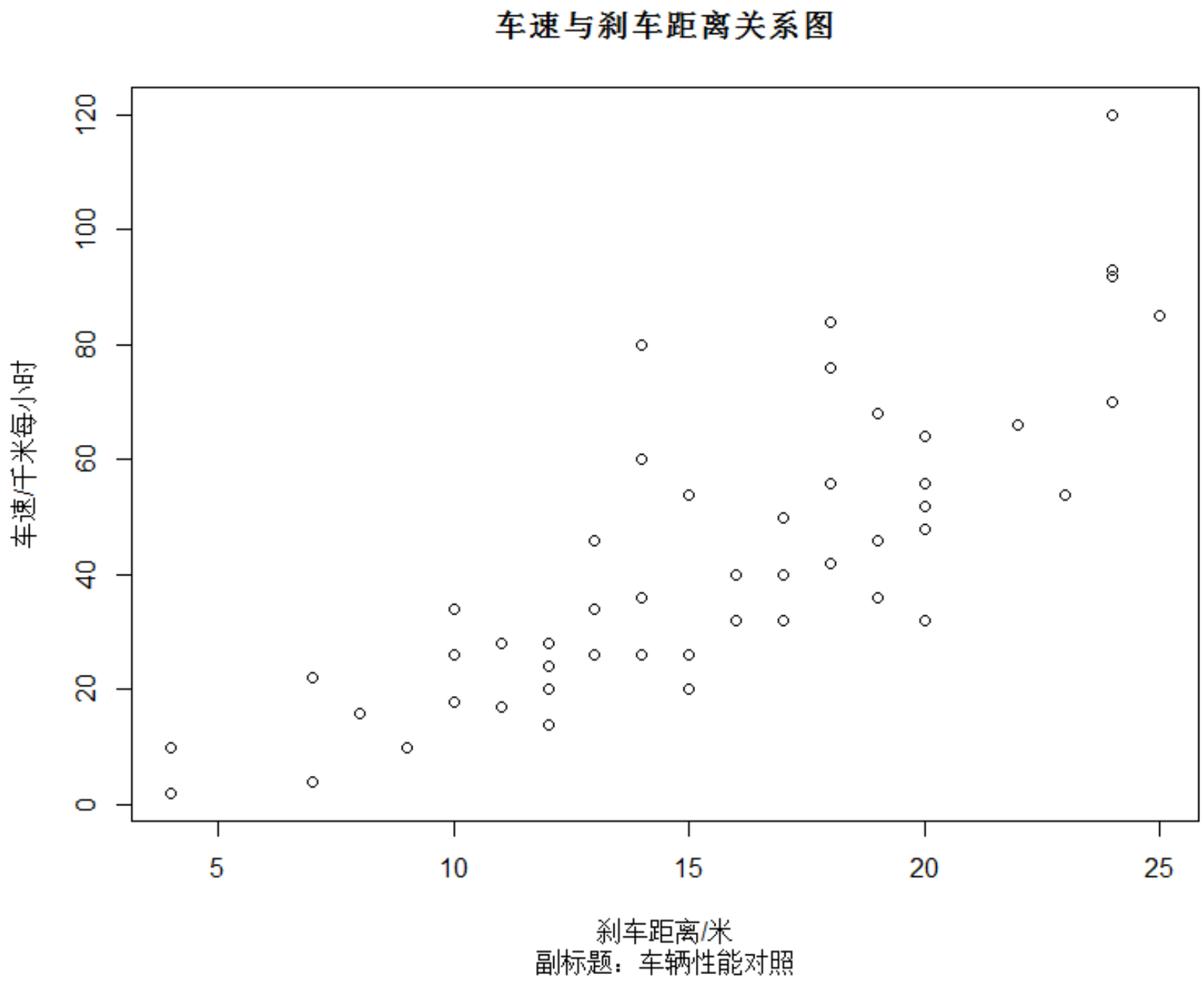
右侧图中将cars数据集使用散点图进行了展示

你能够找出多少图形元素?
观察车速与刹车距离关系图变化时plot中参数的变化

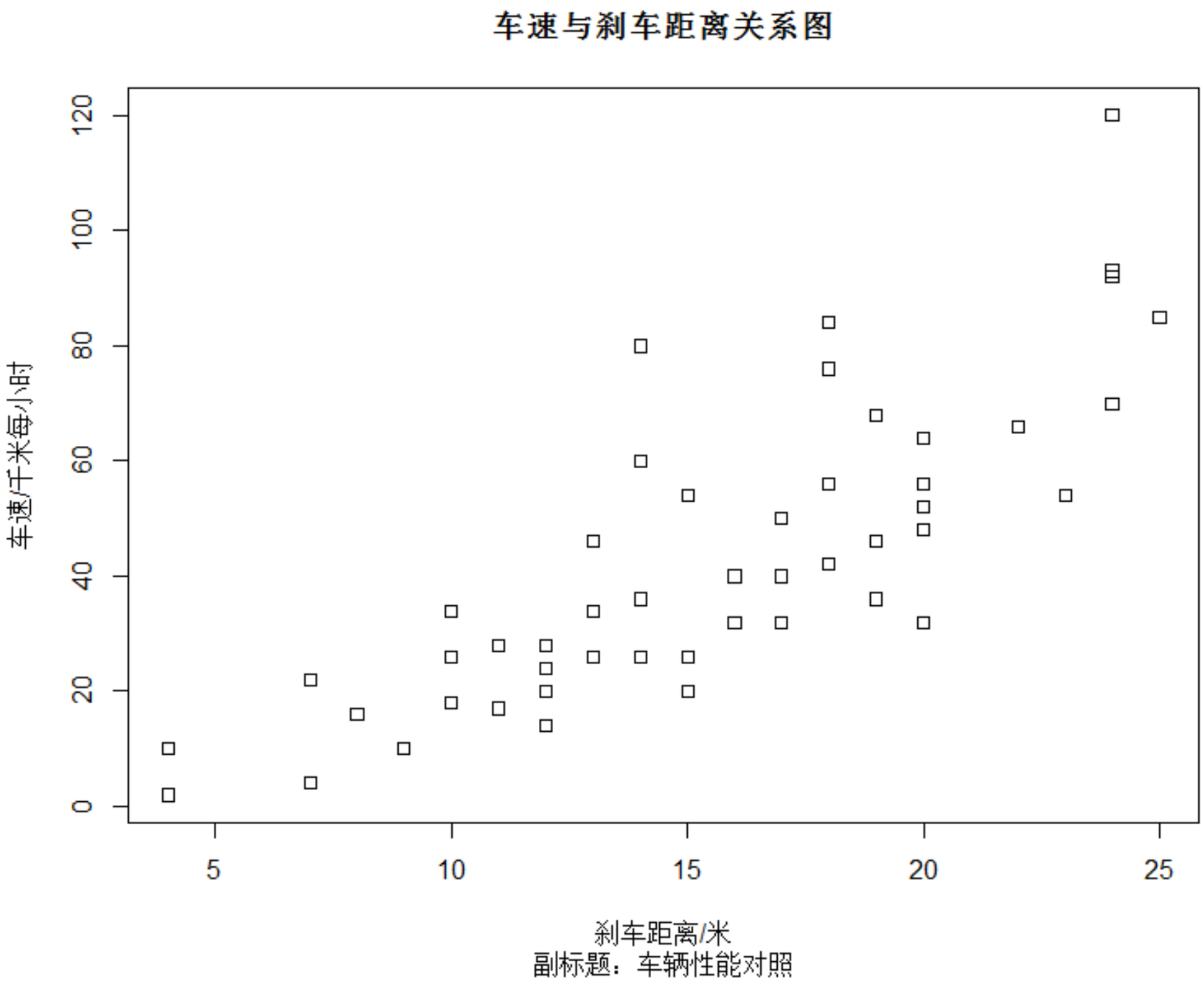



```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时")
```

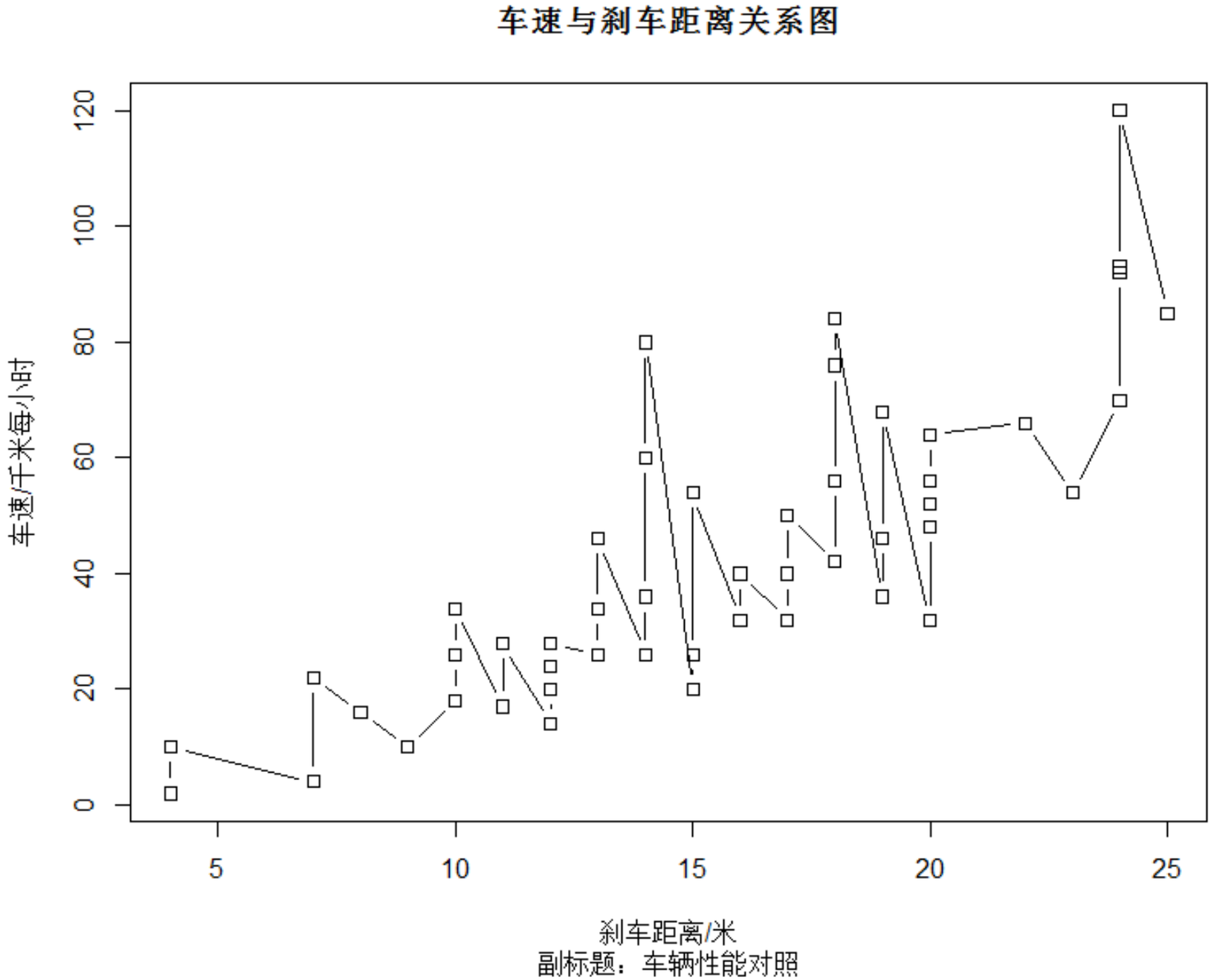
```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照")
```



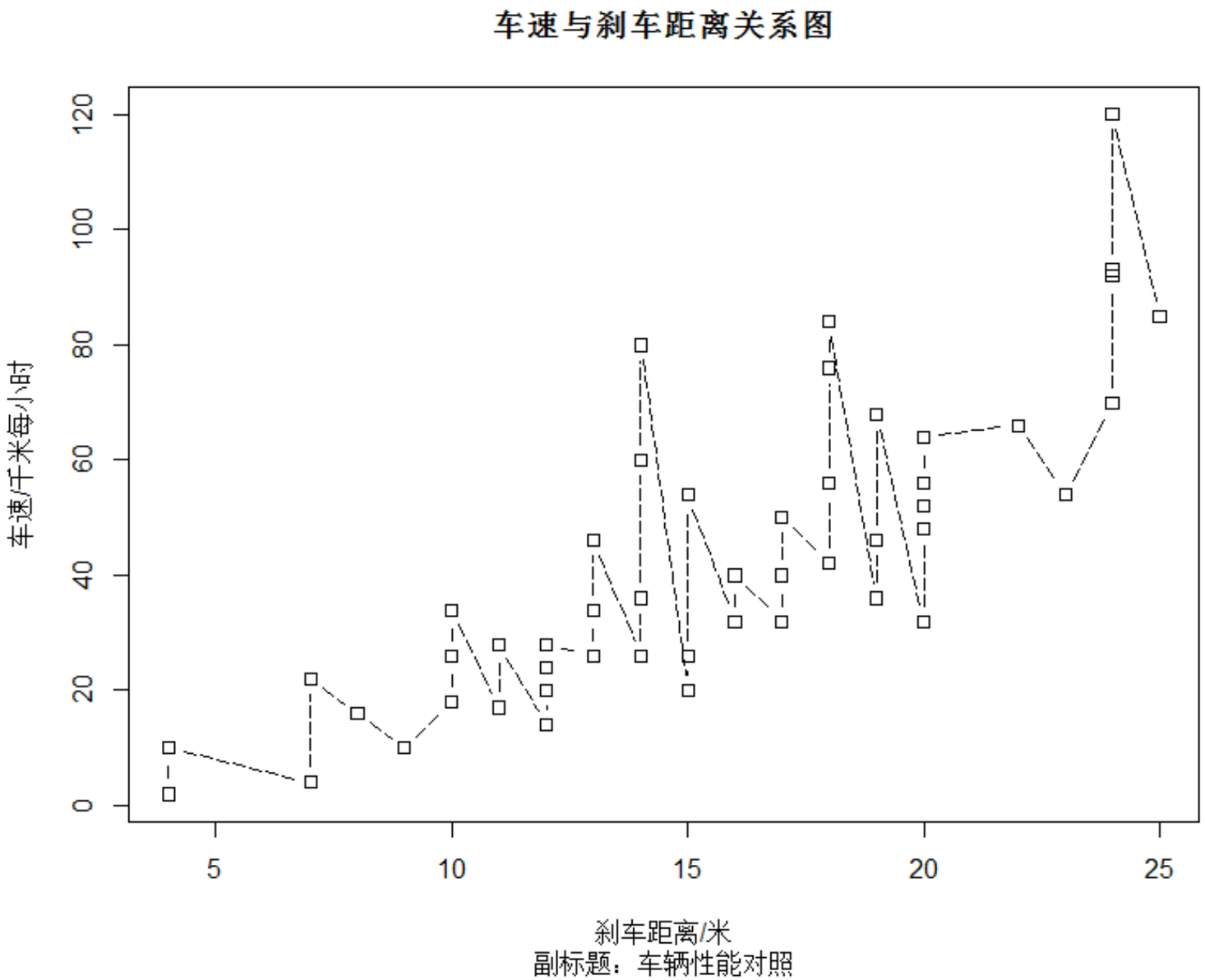
```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照",pch=0)
```



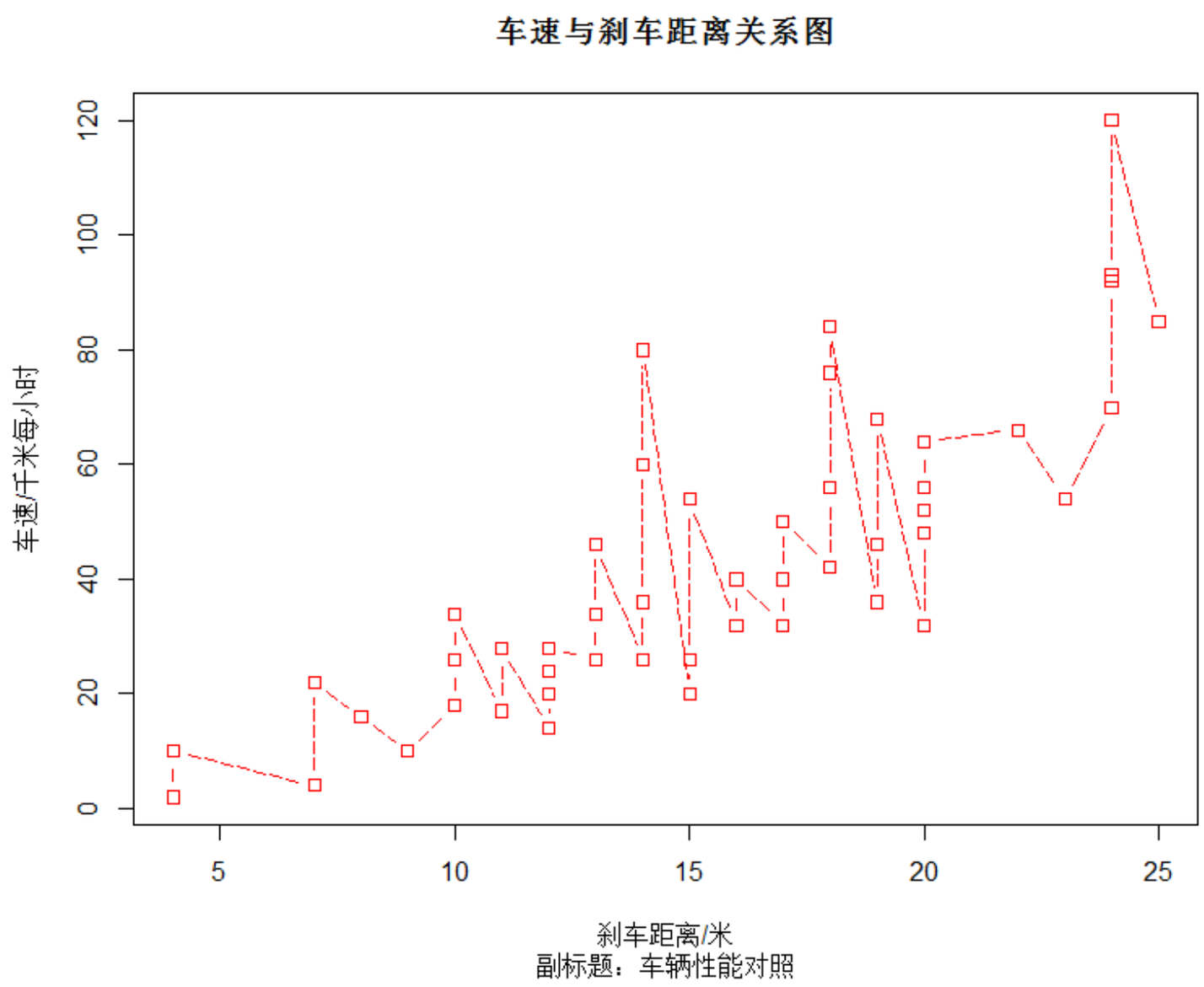
```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照"
,pch=0,type="b")
```



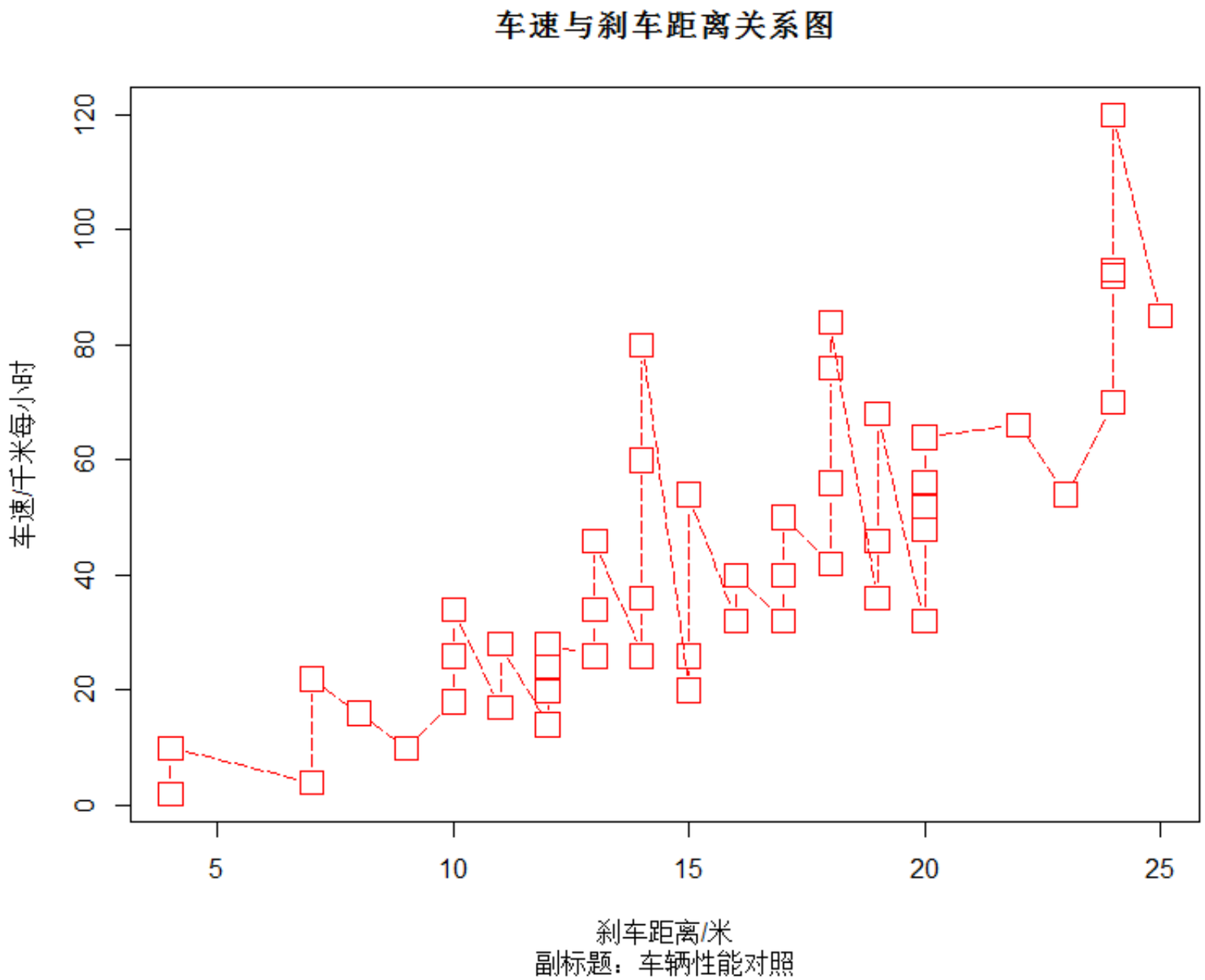
```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照"
,pch=0,type="b",lty=5)
```



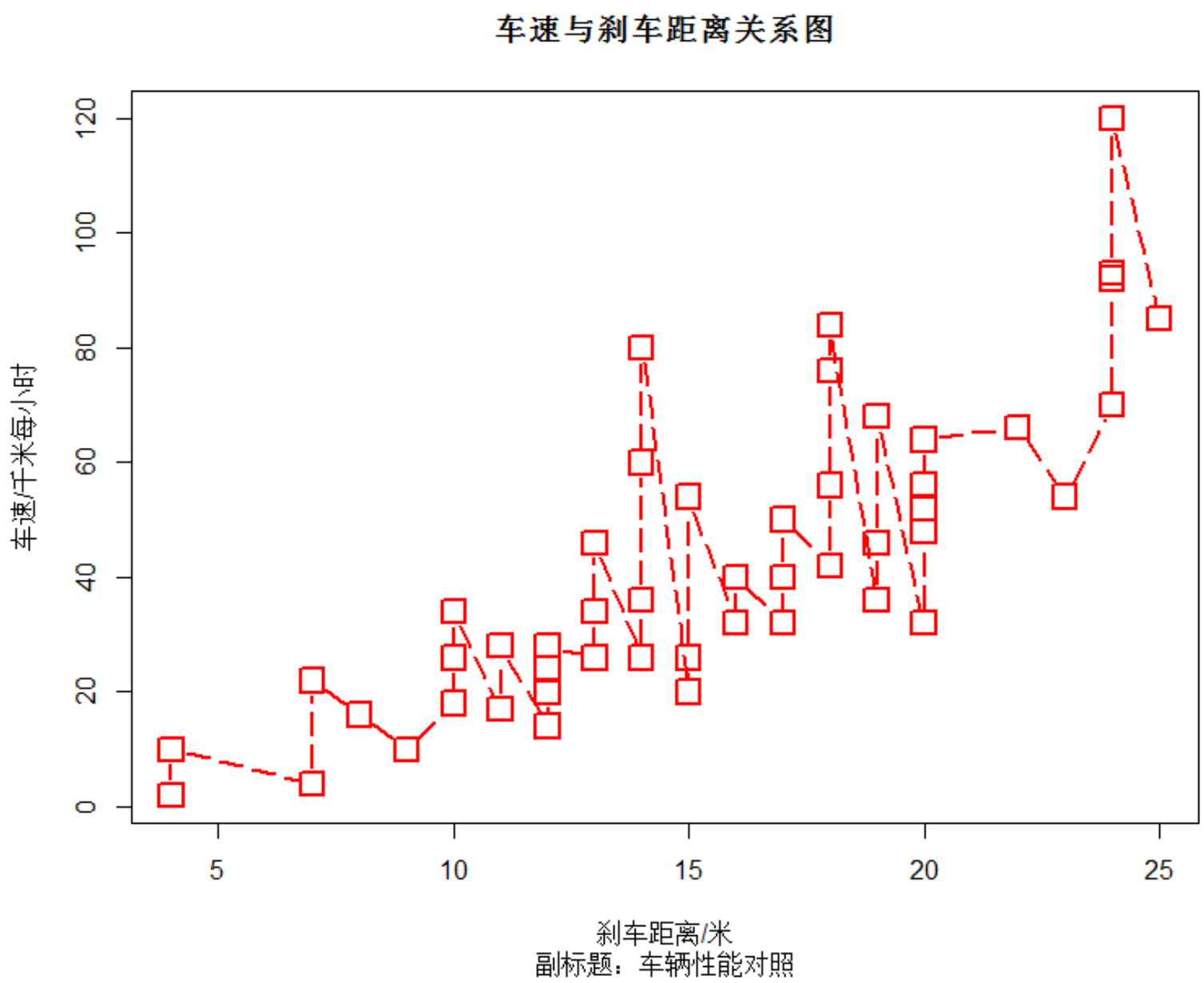
```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照"
,pch=0,type="b",lty=5,col="red")
```



```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照
",pch=0,type="b",lty=5,col="red",cex=2)
```

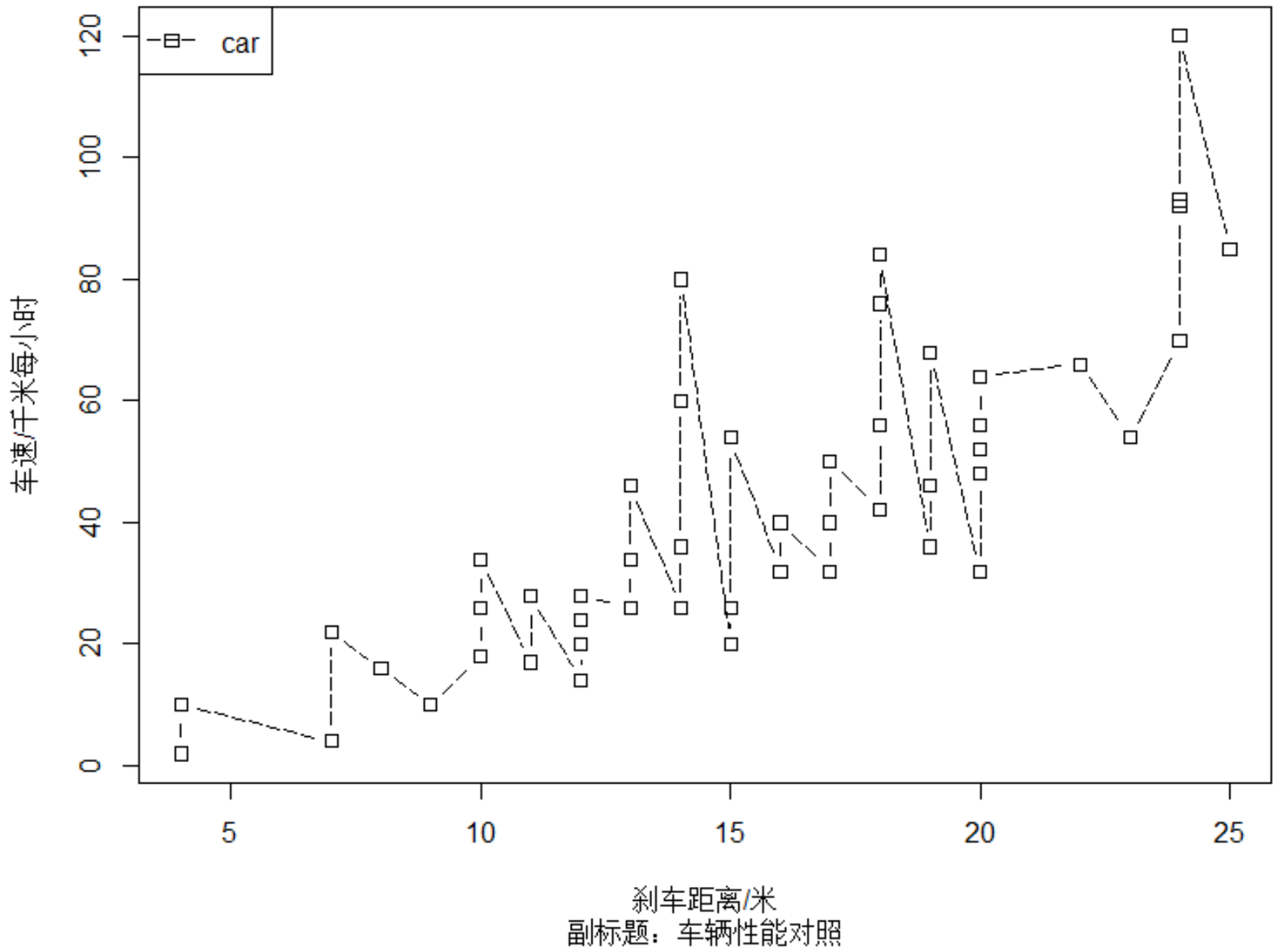


```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照"
,pch=0,type="b",lty=5,col="red",cex=2,lwd=2)
```

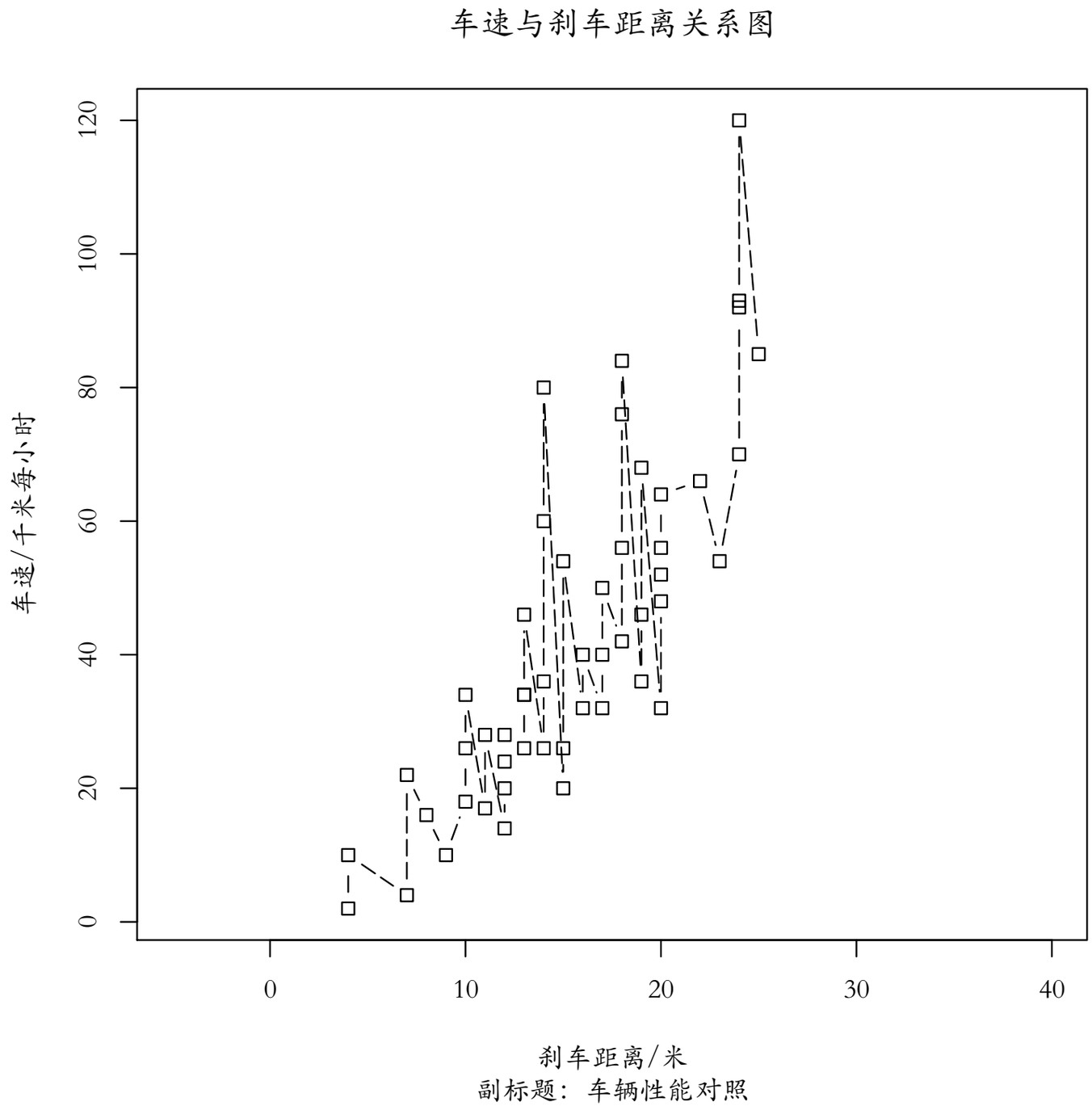


```
> plot(cars,main="车速与刹车距离关系图",xlab="刹车距离/米",ylab="车速/千米每小时",sub="副标题：车辆性能对照"
,pch=0,type="b",lty=5)
```

```
> legend("topleft",legend = "car",lty = 5,pch = 0)
```

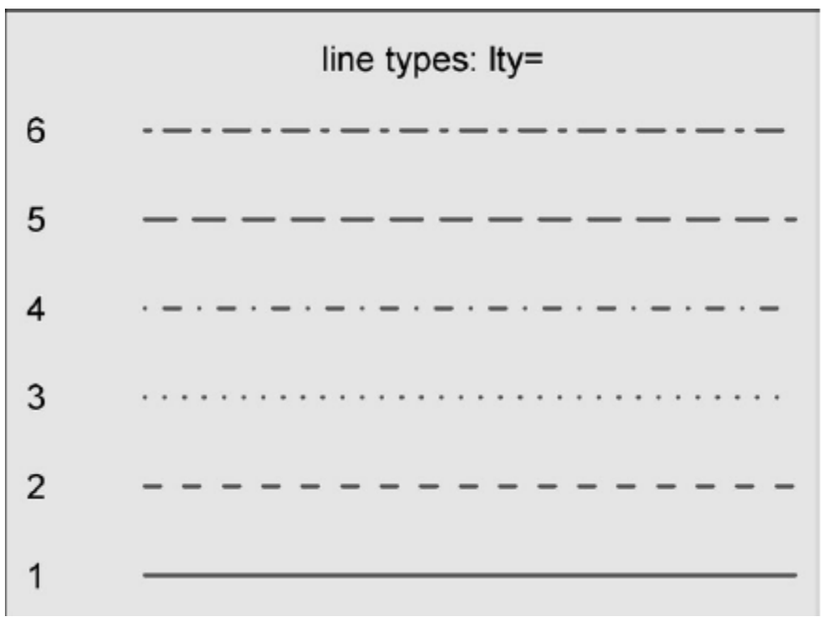


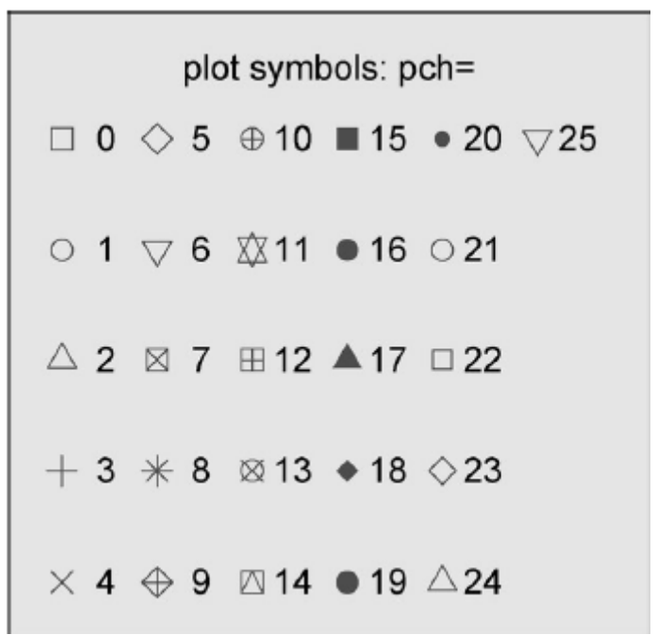
> plot(cars, main="车速与刹车距离关系图", xlab="刹车距离/米", ylab="车速/千米每小时",sub="副标题：车辆性能对照" ,pch=0, type="b", lty=5, xlim=c(-5,40))



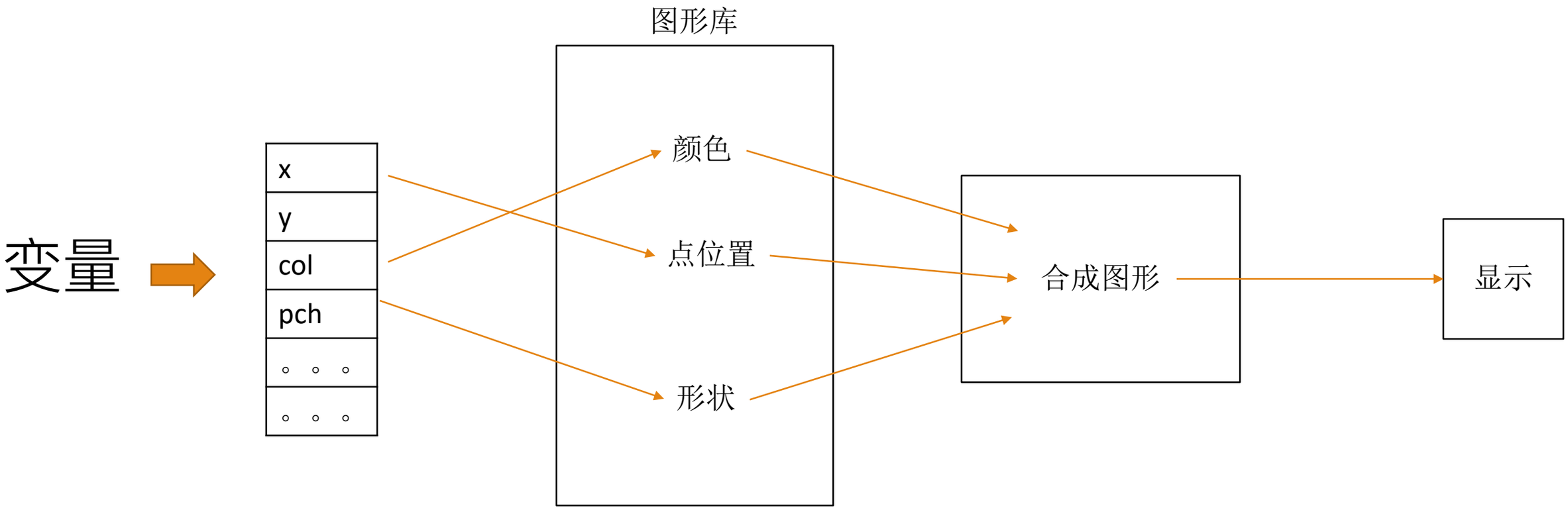
总结:plot中的图形元素名称

```
plot( x,y ,
        type
        pch
        cex
        lty
        lwd
        main
        sub
        xlab
        ylab
        xlim
        ylim )
```





数据可视化过程

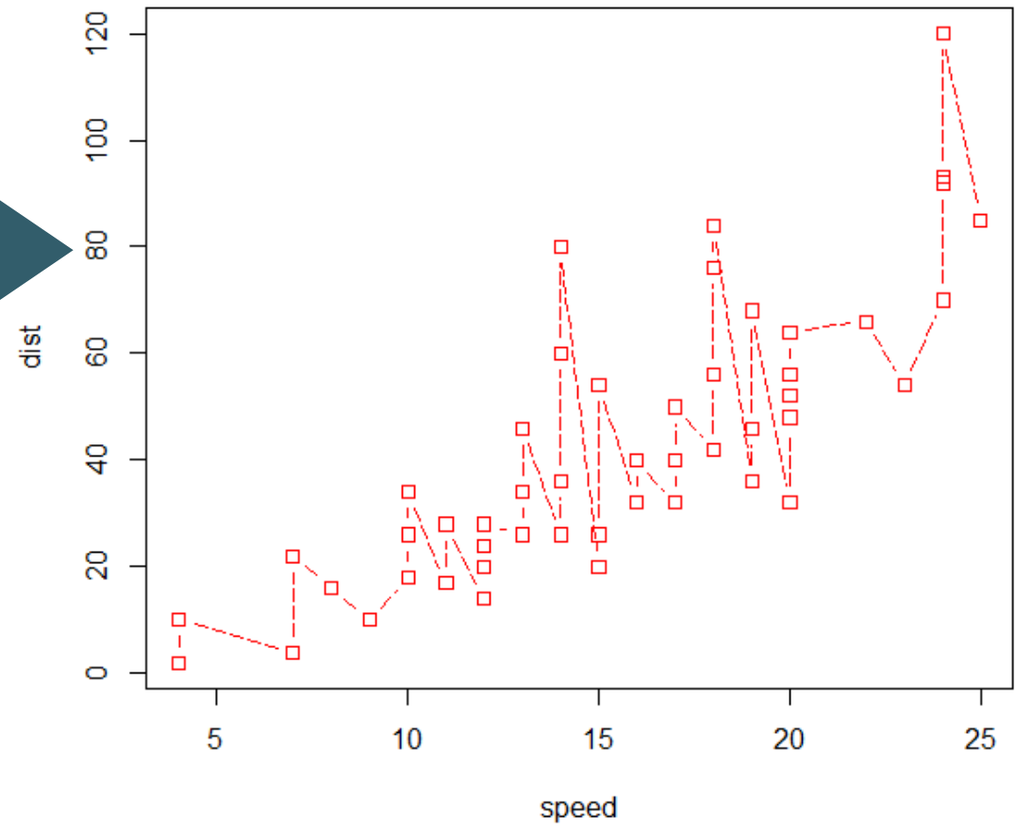


数据与图形元素的映射关系

将要展示的变量映射到x y两个位置属性上

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
7	10	18
8	10	26
9	10	34
10	11	17
11	11	28
12	12	14

x	y	col	pch	lty
4	2	red	0	5
4	10	red	0	5
7	4	red	0	5
7	22	red	0	5
8	16	red	0	5
9	10	red	0	5
10	18	red	0	5




```
> plot(cars$speed,cars$dist,col=2)
```

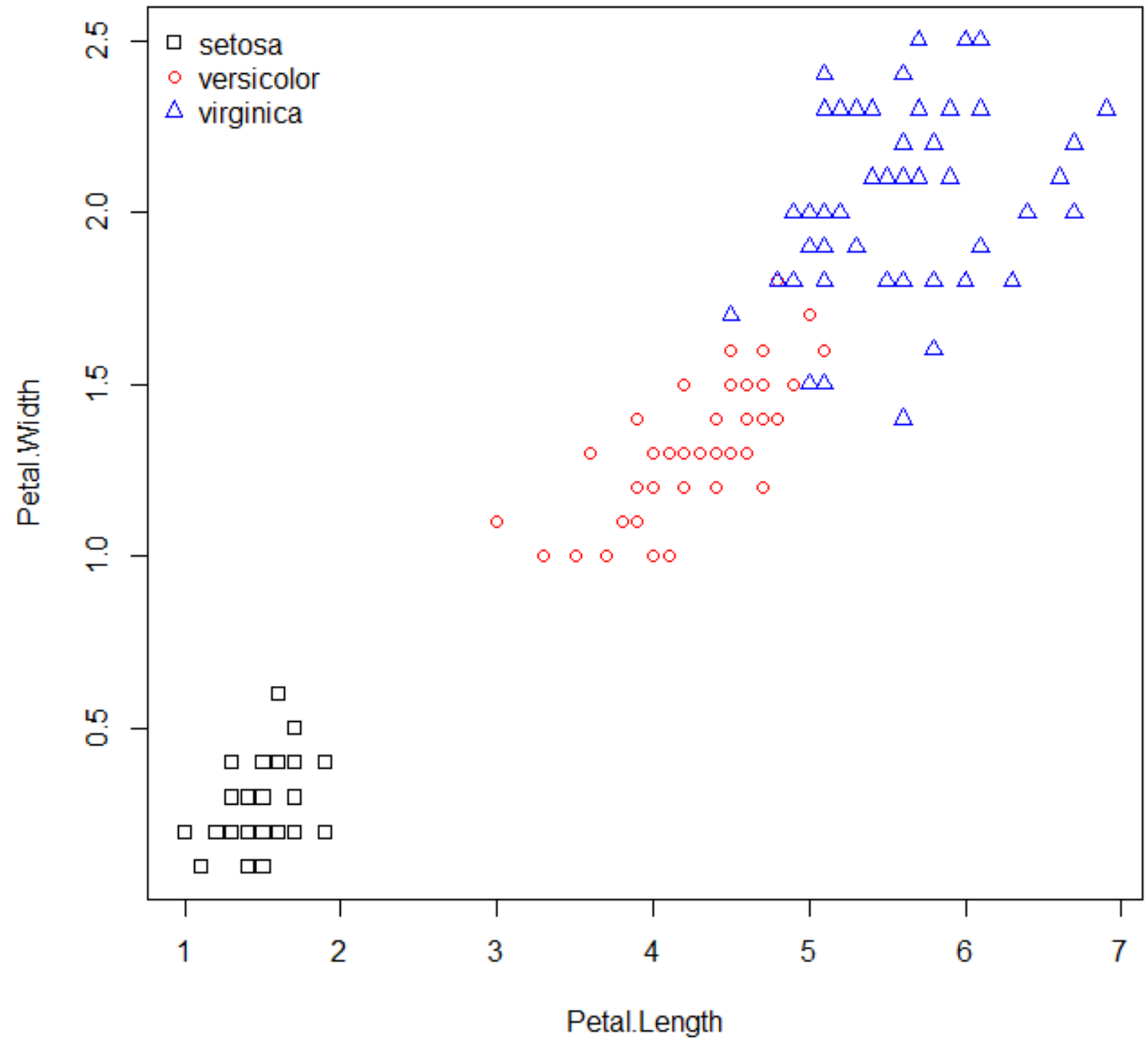
手动为颜色属性进行标度

数据集： iris

```
> str(iris[,3:5])
'data.frame':   150 obs. of  3 variables:
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..:
```

对iris后三列变量进行可视化

第三维度画在了哪里?



	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1			
11	5.4	3.7			
12	4.8	3.4			

变量映射到元素

x	y	col	pch	…
4	2	setosa	0	
4	10	setosa	0	
7	4	setosa	0	
7	22	setosa	0	
8	16	setosa	0	
9	10	setosa	0	
10	18	setosa	0	

标度为col属性

可读的属性值

x	y	col	pch	…
4	2	red	0	
4	10	red	0	
7	4	red	0	
7	22	red	0	
8	16	red	0	
9	10	red	0	
10	18	red	0	

```
> dx<-as.integer(iris$Species)
```

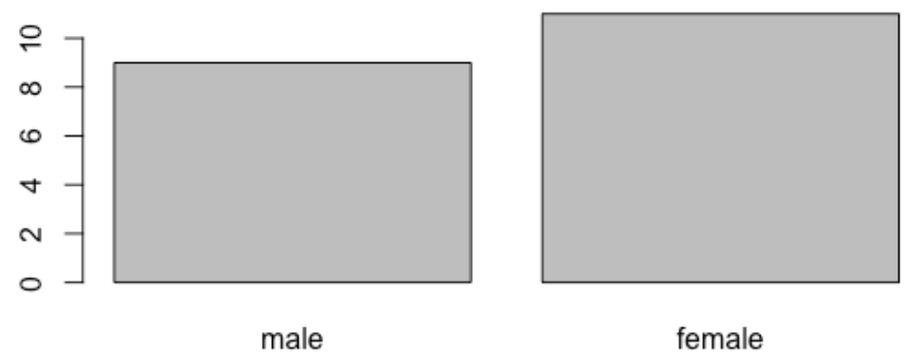
```
> plot(iris[,3:4],pch=c(0,1,2)[dx],col=c("black","red","blue")[dx])
```

其他图形函数

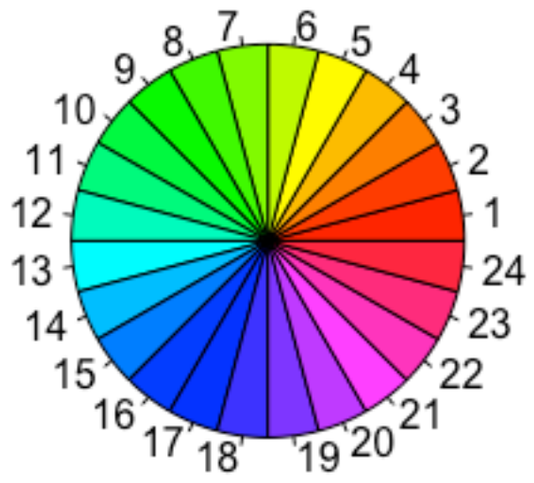
plot是最为常用的做图函数，此外，其他同类的做图函数还有：

barplot
boxplot
pie
hist
qqnorm

这些图形函数相当于一套模版，我们要做的工作就是建立好变量与图形参数之间映射关系，并表达给计算机。




```
> barplot(c(nrow(male),nrow(female)),names.arg  =c('male','female'))
```




```
> pie(rep(1, 24), col = rainbow(24), radius = 0.9)
```

添加元素

添加图例

cars数据集：单变量

```
> legend('topleft',lty = 1,pch=1,legend = 'cars')
```

```
> legend("topleft",inset=0.03,legend="car",lty=1,pch=1,col="blue",bty="n")
```

iris数据集：3个变量

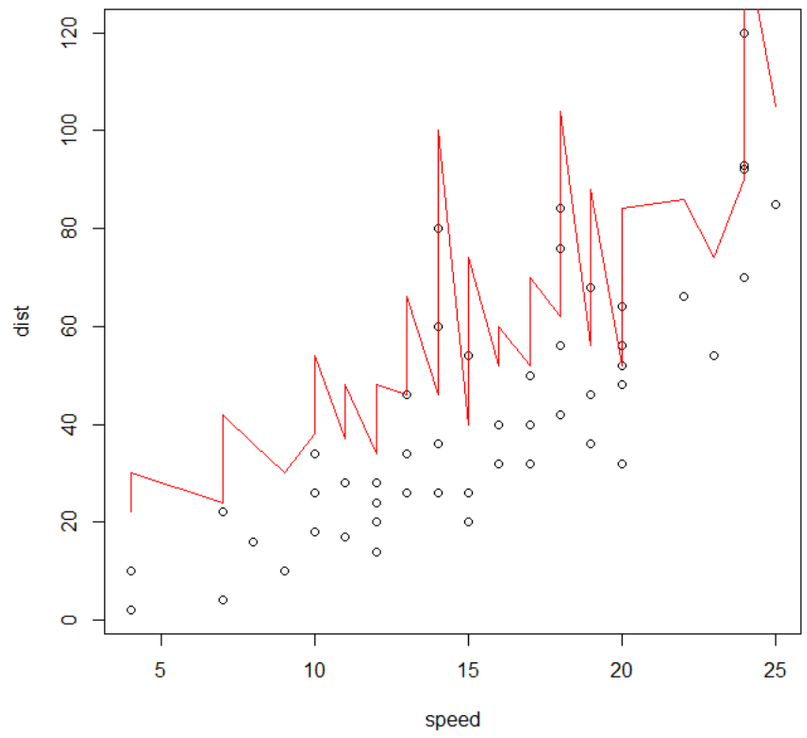
```
> dx<-as.integer(iris$Species)
```

```
> plot(iris[,3:4],pch=c(0,1,2)[dx],col=c("black","red","blue")[dx])
```

```
> legend("topleft",legend=levels(iris$Species),pch=c(0,1,2),col=c("black","red","blue"),bty="n")
```

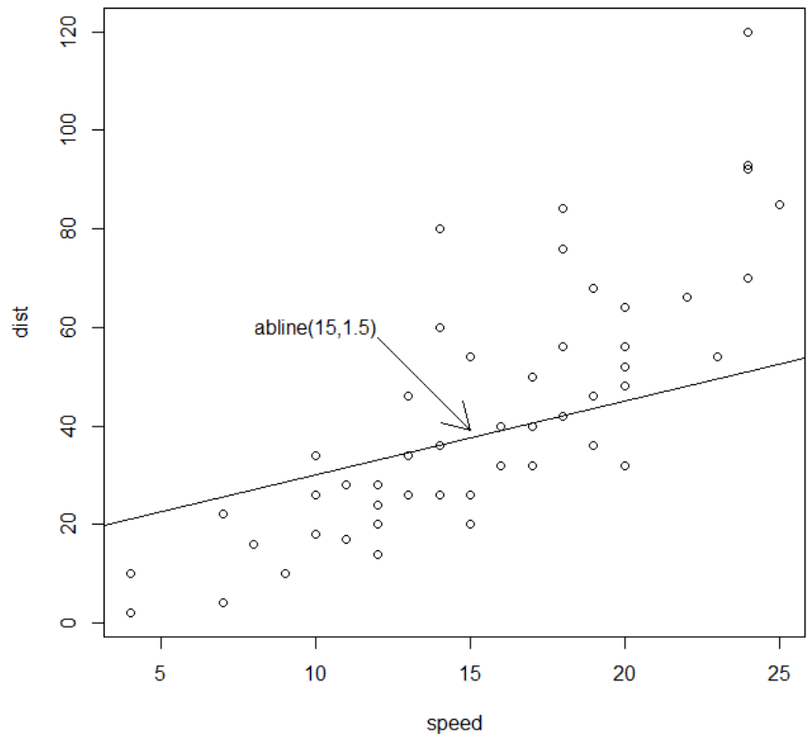
添加线　相当于plot当中的type="l"类型

```
> lines(cars$speed,cars$dist+20,col="red")
```



添加斜率-截距线：abline

```
> plot(cars)
> abline(15,1.5)
> text(10,60,labels="abline(15,1.5)")
> arrows(10,58,15,39)
```




```
> abline(h=60)
```

h参数代表水平线

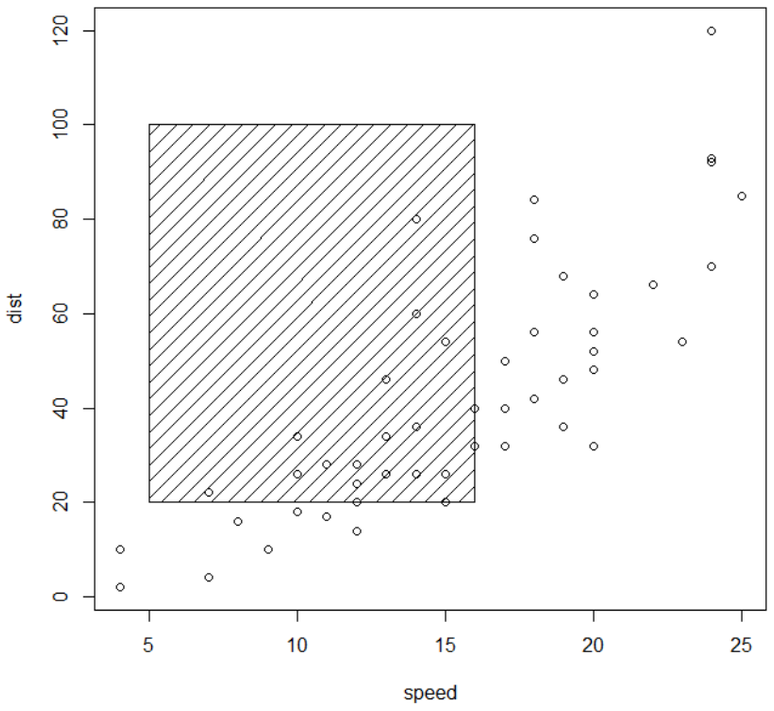
```
> abline(v=20,lty=2)
```

v参数代表垂直线

text仅能添加坐标轴内文本，
按坐标定位

添加矩形

```
> rect(5,20,16,100,density=10)
```



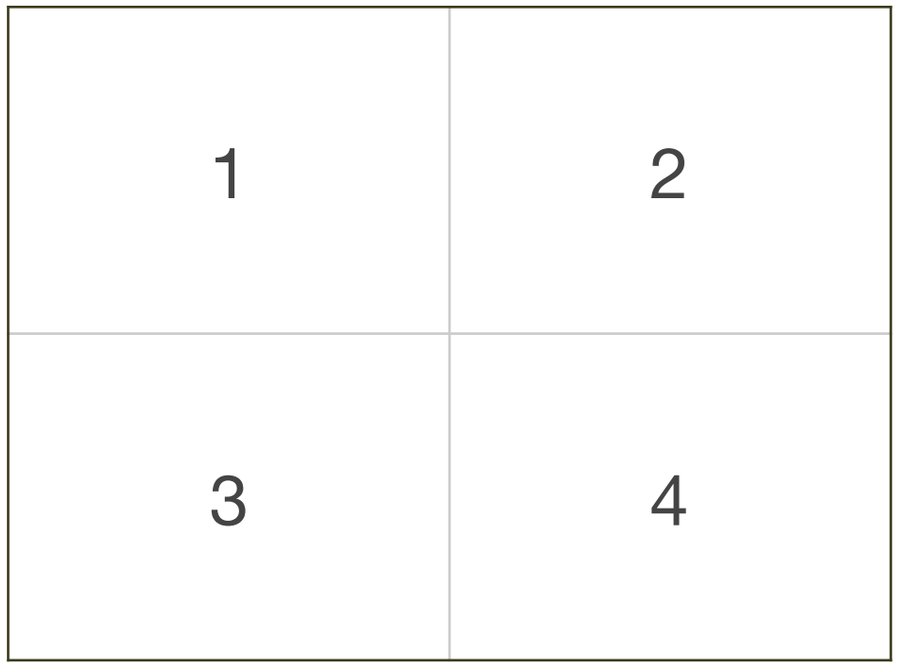
title( )	添加标题
text( )	在绘图区添加文字
mtext( )	在边界区添加文字
points( )	添加点
lines( )	添加线
abline( )	添加参考线
axis	添加坐标轴
legend	添加图例
polygon	添加多边形

分面布局

图形界面的分割需要在设备默认参数里操作，即通过par()

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

将图形窗口分割为2X2的四个子图
在随后的做图过程中，每画一个图就按右侧示意图中的编号顺序摆放。

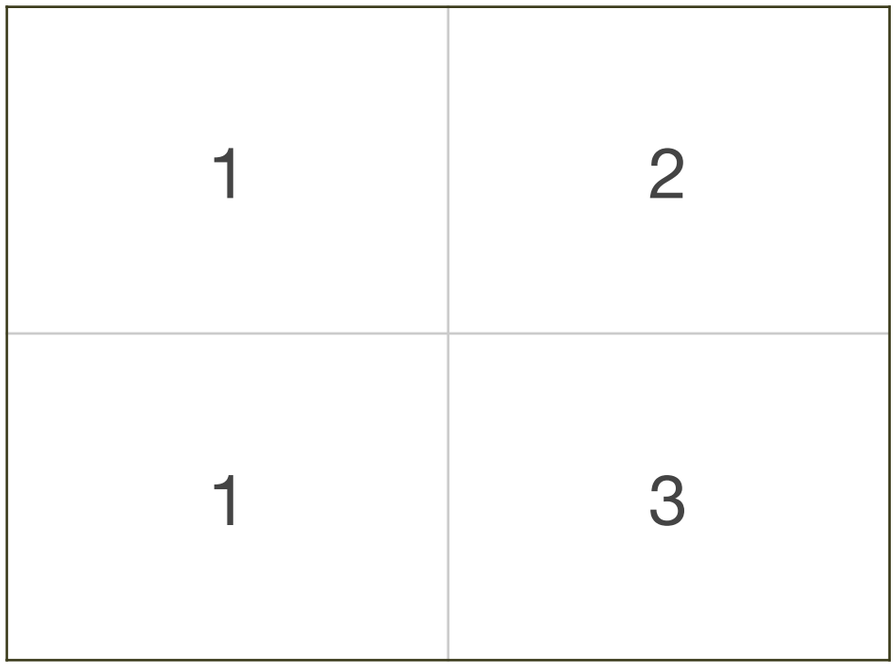


尝试下列代码会生成什么样的图形

```
> par(mfrow=c(3,1))
> hist(mtcars$wt)
> hist(mtcars$mpg)
> hist(mtcars$disp)
```

R利用将图形界面等分成若干行列的田字格，利用矩阵表示分面的序号。
我们可以更灵活的将部分分面合并，任意构造出分面
如下操作，layout函数将一个写好顺序编号的矩阵替代默认分割方式

```
> layout(matrix(c(1,1,2,3),nrow=2))
> hist(mtcars$wt)
> hist(mtcars$mpg)
> hist(mtcars$disp)
```



尝试代码输出效果