The R Book

Chapter 2: Essentials of the R Language
Session 9

Writing functions in R

objects that carry out operations on arguments

Syntax:
function (argument « variable ») body

Keyword: **function** indicates to R that you want to create a function.

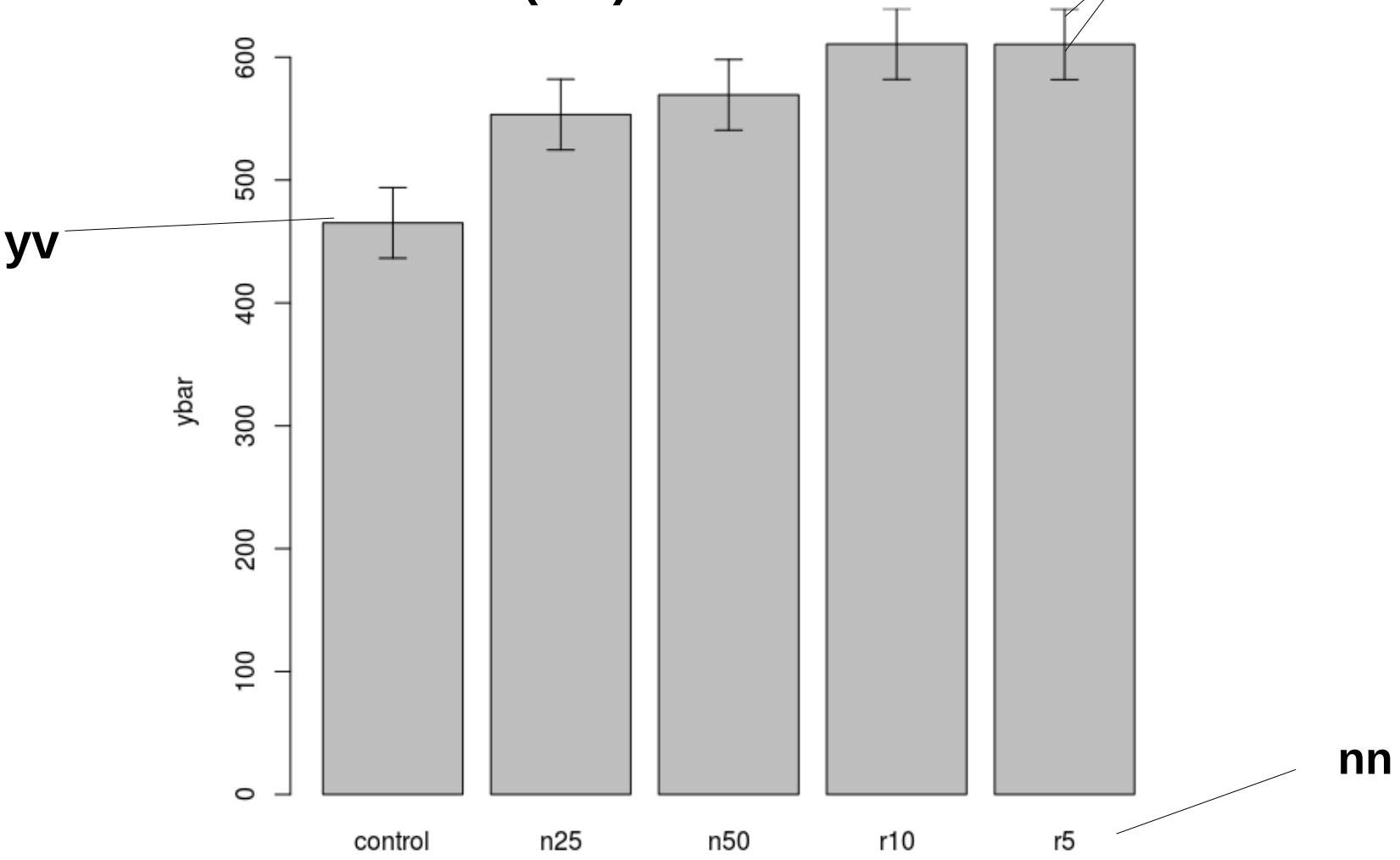
Argument - comma-separated list of formal arguments.
- variable symbol (x or y),
- symbol = expression (pch=16)
- ...

Body - any valid R expression / set of R expressions.
- often contained in curly brackets { }, each expression on a separate line.

Error Bars

Write function with arguments:

- **heights of bars, (yv)**
- **lengths (up and down) of error bars, z**
- **labels for bars (nn)**

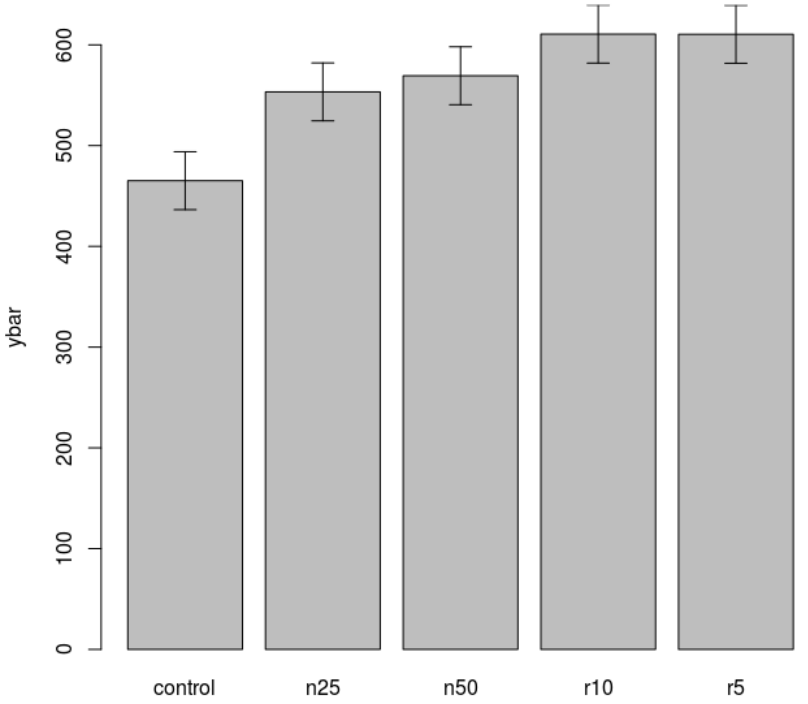


Error Bars

Write function with arguments:

- heights of bars, (yv)
- error bar length (z)
- labels for bars (nn)

```
> error.bars<-function(yv,z,nn){
```



value

y-axis

y-labels

```
+ xv<-barplot(yv,ylim=c(0,(max(yv)
+ +max(z))),names=nn,ylab=deparse(substitute(yv)
+ ))
+ ...
```

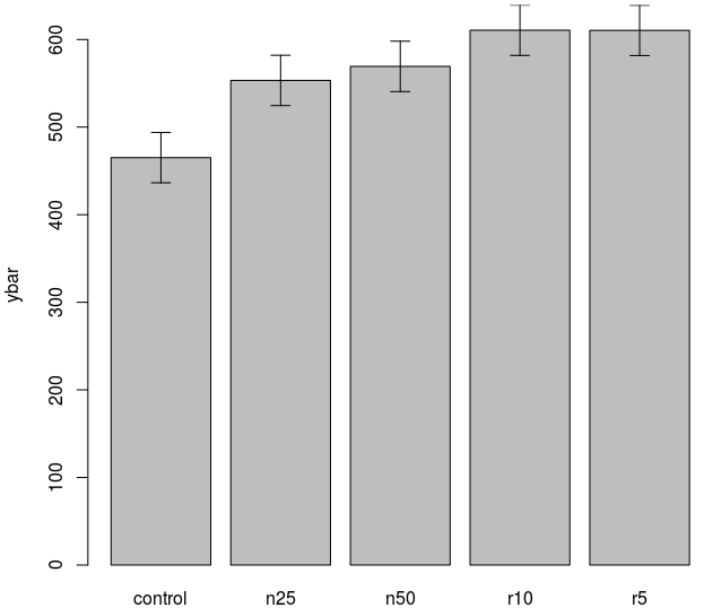
1

x-labels

expresion into a character string

returns the parse tree not its content ~name of variable

Error Bars



Write function with arguments:

- heights of bars, (yv)
- error bar length (z)
- labels for bars (nn)

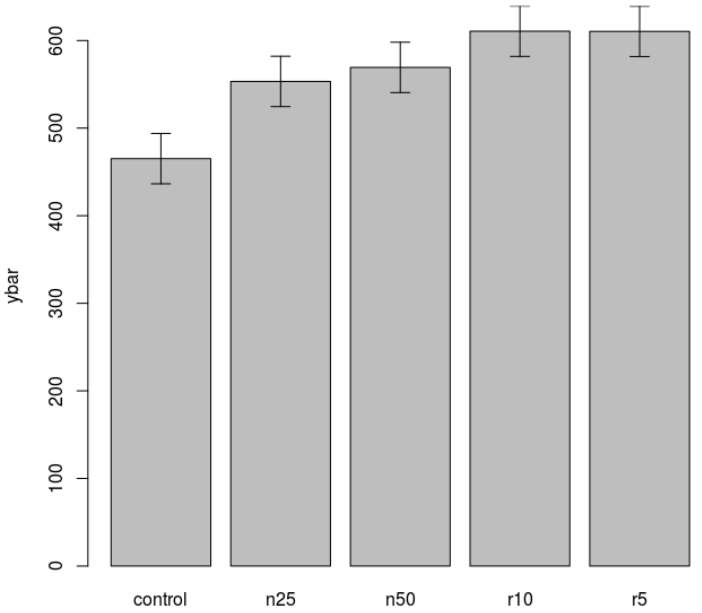
```
> error.bars<-function(yv,z,nn){

+ xv<-barplot(yv,ylim=c(0,(max(yv)
+ +max(z))),names=nn,ylab=deparse(substitute(yv)
+ ))

+ g=(max(xv)-min(xv))/50
```

1

2

proportional length of horizontal error bars

Error Bars



Write function with arguments:

- heights of bars, (yv)
- error bar length (z)
- labels for bars (nn)

```
> error.bars<-function(yv,z,nn){

+ xv<-barplot(yv,ylim=c(0,(max(yv)
+ +max(z))),names=nn,ylab=deparse(substitute(yv)
+ ))

+ g=(max(xv)-min(xv))/50

+ for (i in 1:length(xv)) {
+ lines(c(xv[i],xv[i]),c(yv[i]+z[i],yv[i]-z[i]))
+ lines(c(xv[i]-g,xv[i]+g),c(yv[i]+z[i], yv[i]+z[i]))
+ lines(c(xv[i]-g,xv[i]+g),c(yv[i]-z[i], yv[i]-z[i]))
+ }}
```

1

2

3

Allocating the lines

vertical

upper horizontal

lower horizontal

Error Bars

Use your function …

```
> comp<-read.table(/.../competition.txt",header=T)
> attach(comp)
> names(comp)
[1] "biomass"  "clipping"

> comp
...
```

Error Bars

Use your function …

```
> comp<-read.table(/.../competition.txt",header=T)
> attach(comp)
> names(comp)
[1] "biomass"  "clipping"

> comp
…
> se<-rep(28.75,5)
> labels<-as.character(levels(clipping))
> ybar<-as.vector(tapply(biomass,clipping,mean))
```

Statistics calculator values categories stats

```
> error.bars(ybar,se,labels)
```

… what do you observe ???

Exercise :

- Change error bar width and length
- Incorporate the standard error of each sample

Error Bars

Task : plot the standard error of each sample

Help:

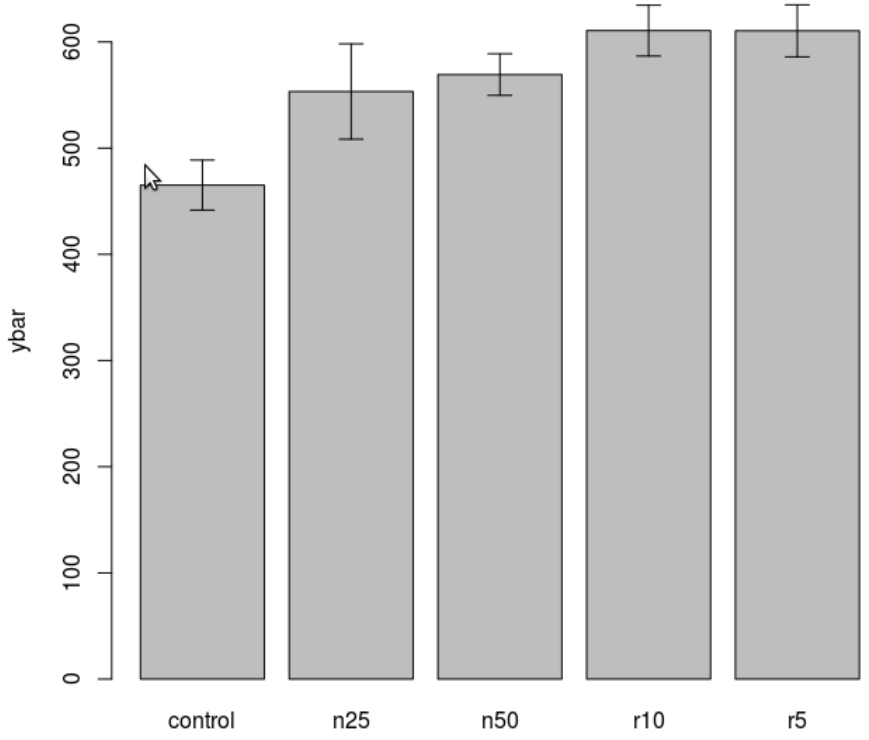
- `std.error<-function(x) sqrt(var(x)/length(x))`
- `tapply(biomass,clipping, ...)`

Error Bars

Task : plot the standard error of each sample

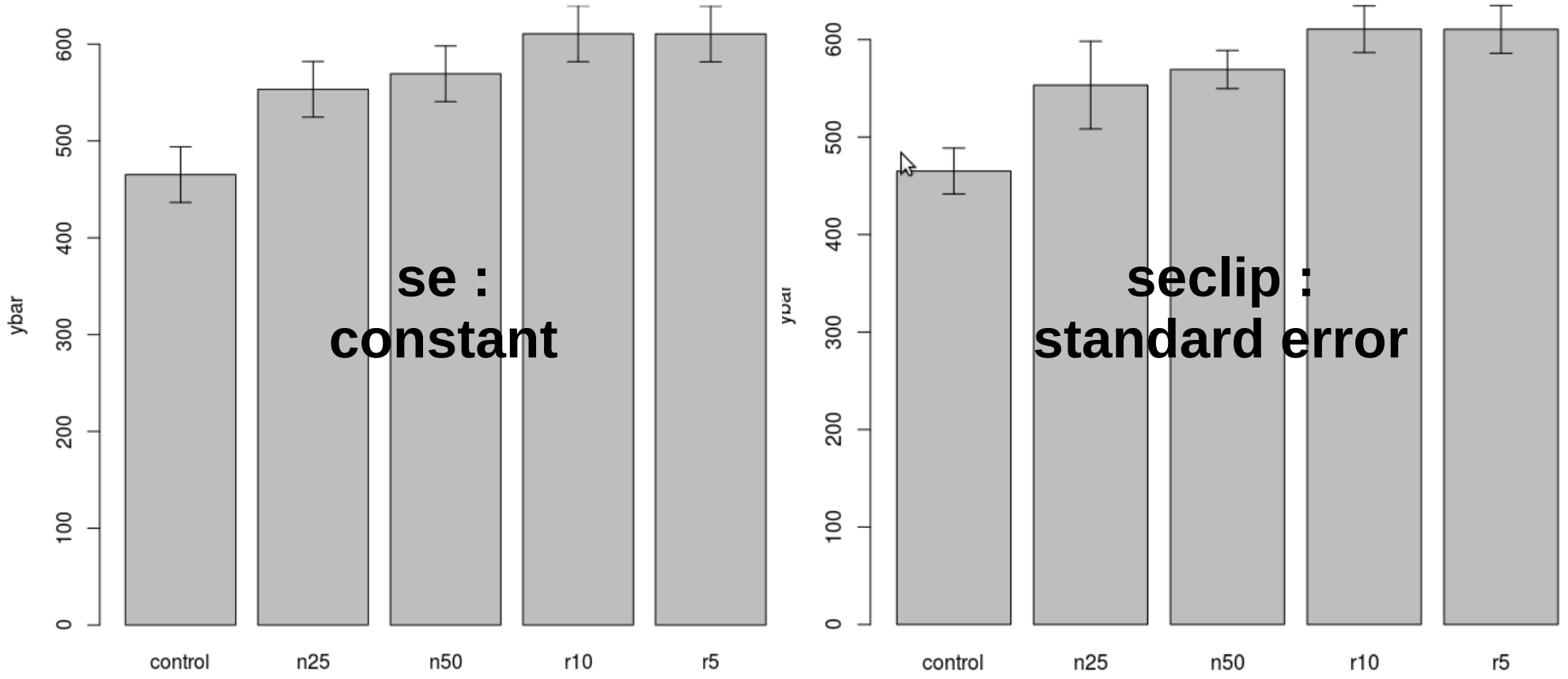
- std.error<-function(x) sqrt(var(x)/length(x))
- tapply(biomass,clipping, ...)

```
> seclip<-as.vector(sqrt(tapply(biomass,clipping,var)/
+ tapply(biomass,clipping,length)))
> labels<-as.character(levels(clipping))
> ybar<-as.vector(tapply(biomass,clipping,mean))
> error.bars(ybar,seclip,labels)
```



Error Bars

```
> seclip<-as.vector(sqrt(tapply(biomass,clipping,var)/
+ tapply(biomass,clipping,length)))
> labels<-as.character(levels(clipping))
> ybar<-as.vector(tapply(biomass,clipping,mean))

> error.bars(ybar,seclip,labels)
```



error bars on a scatterplot in both the x and y directions:

```
> xy.error.bars<-function (x,y,xbar,ybar){
+ plot(x, y, pch=16, ylim=c(min(y-ybar),max(y+ybar)),
```

Point character

What is that ???

```
+ xlim=c(min(x-xbar),max(x+xbar))
```

...

error bars on a scatterplot in both the x and y directions:

```
> xy.error.bars<-function (x,y,xbar,ybar){
+ plot(x, y, pch=16, ylim=c(min(y-ybar),max(y+ybar)),
```

Point character

What is that ???

```
+ xlim=c(min(x-xbar),max(x+xbar)))
```

x(0) y(0) x(1) y(1)

arrow angle shaft /head

edge length

```
+ arrows(x, y-ybar, x, y+ybar, code=3, angle=90, length=0.1)
```

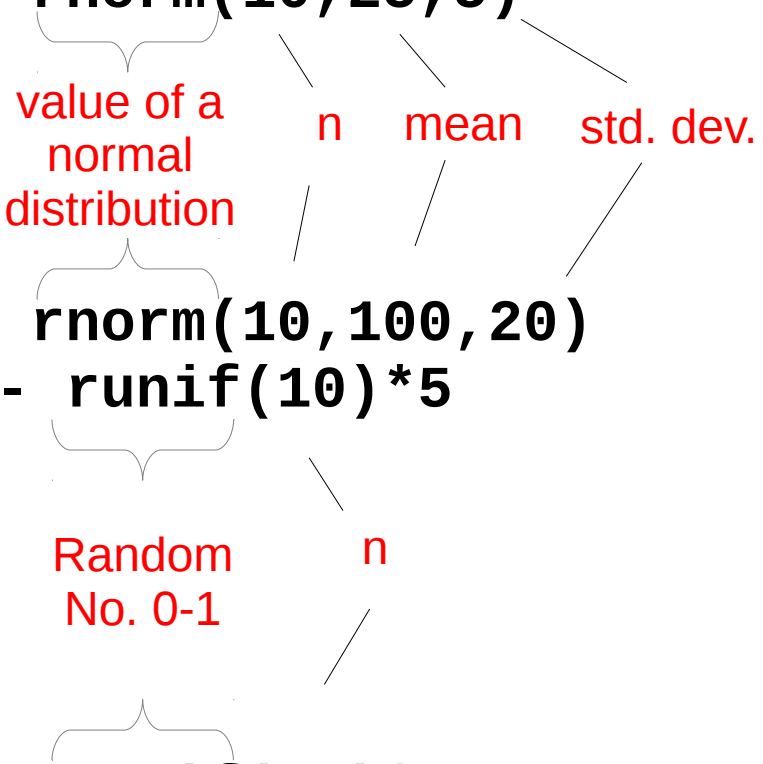
draws arrows between pairs of points.

kind of arrows

```
+ arrows(x-xbar, y, x+xbar, y, code=3, angle=90, length=0.1)}
```

error bars on a scatterplot in both the x and y directions:

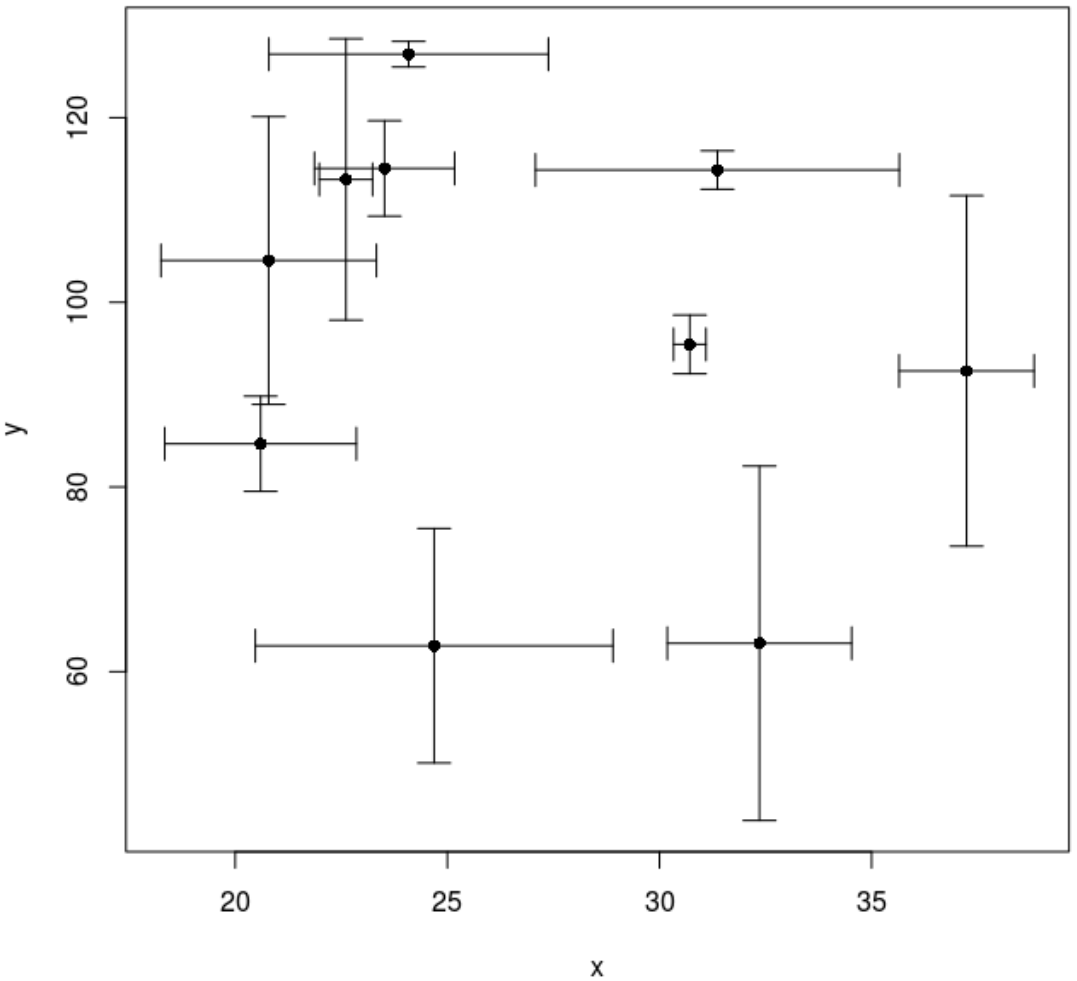
```
> xy.error.bars<-function (x,y,xbar,ybar){
+ plot(x, y, pch=16, ylim=c(min(y-ybar),max(y+ybar)),
+ xlim=c(min(x-xbar),max(x+xbar)))
+ arrows(x, y-ybar, x, y+ybar, code=3, angle=90, length=0.1)
+ arrows(x-xbar, y, x+xbar, y, code=3, angle=90, length=0.1)}

> x <- rnorm(10,25,5)
> y <- rnorm(10,100,20)
> xb <- runif(10)*5
> yb <- runif(10)*20
> xy.error.bars(x,y,xb,yb)
```



error bars on a scatterplot in both the x and y directions:

```
> xy.error.bars<-function (x,y,xbar,ybar){
+ plot(x, y, pch=16, ylim=c(min(y-ybar),max(y+ybar)),
+ xlim=c(min(x-xbar),max(x+xbar)))
+ arrows(x, y-ybar, x, y+ybar, code=3, angle=90, length=0.1)
+ arrows(x-xbar, y, x+xbar, y, code=3, angle=90, length=0.1)}
```

```
> x <- rnorm(10,25,5)
> y <- rnorm(10,100,20)
> xb <- runif(10)*5
> yb <- runif(10)*20

> xy.error.bars(x,y,xb,yb)
```



error bars on a scatterplot in both the x and y directions:

```
> xy.error.bars<-function (x,y,xbar,ybar){
+ plot(x, y, pch=16, ylim=c(min(y-ybar),max(y+ybar)),
+ xlim=c(min(x-xbar),max(x+xbar)))
+ arrows(x, y-ybar, x, y+ybar, code=3, angle=90, length=0.1)
+ arrows(x-xbar, y, x+xbar, y, code=3, angle=90, length=0.1)}

> x <- rnorm(10,25,5)
> y <- rnorm(10,100,20)
> xb <- runif(10)*5
> yb <- runif(10)*20

> xy.error.bars(x,y,xb,yb)
```

Task

- Change point character
- Change arrow length, angle, code

Homework:

- draw a scatterplot with the data you like

Loops and Repeats

For-loop

```
> for (i in 1:5) print(i^ 2)
 [1] 1
 [1] 4
 [1] 9
 [1] 16
 [1] 25
```

single line

```
> j<-k<-0
> for (i in 1:5) {
+ j<-j+1
+ k<-k+i*j
+ print(i+j+k) }
[1] 3
[1] 9
[1] 20
[1] 38
[1] 65
```

multiple lines

Loops and Repeats

For-loop

```
> for (i in 1:5) print(i^ 2)
 [1] 1
 [1] 4
 [1] 9
 [1] 16
 [1] 25
```

single line

```
> j<-k<-0
> for (i in 1:5) {
+ j<-j+1
+ k<-k+i*j
+ print(i+j+k) }
[1] 3
[1] 9
[1] 20
[1] 38
[1] 65
```

multiple lines