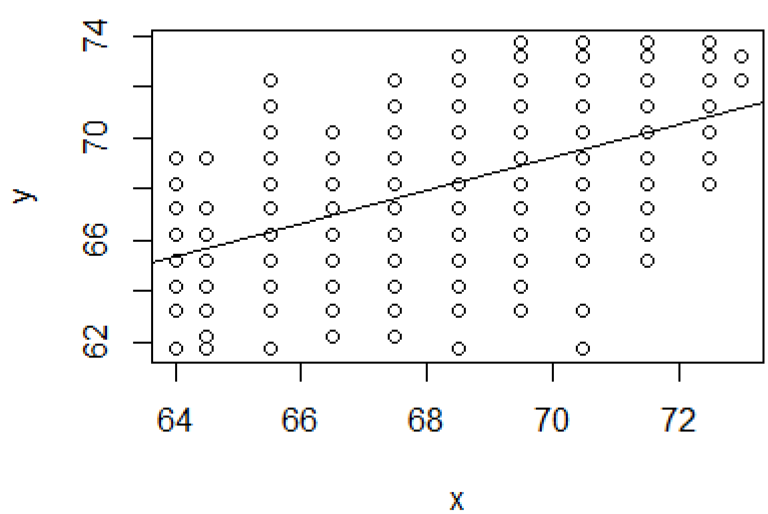
Comparing slopes

Original Data

```
> y <- galton$child
> x <- galton$parent

> beta1 <- cor(y, x) * sd(y) / sd(x)
> beta1
[1] 0.6462906

> beta0 <- mean(y) - beta1 * mean(x)
> beta0
[1] 23.94153

> lm(y ~ x, galton)

Call:
lm(formula = y ~ x, data = galton)

Coefficients:
(Intercept)            x
    23.9415       0.6463

> plot(x,y)
> abline(coef = coef(lm(y~x)))
```

If we wanted a line that goes through the origin (0,0), we can calculate the slope by:

```
> sum(y * x) / sum(x ^ 2)
[1] 0.9965439
```

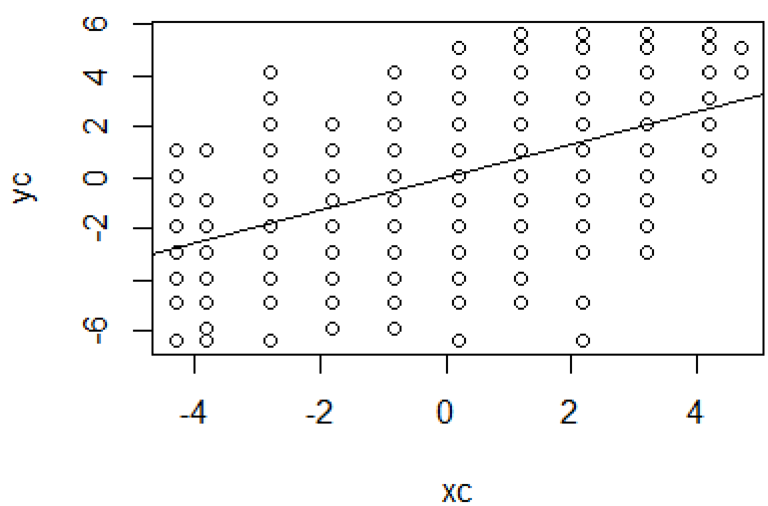


Centered Data

```
> yc <- y - mean(y)
> xc <- x - mean(x)
> beta1c <- cor(yc, xc) * sd(yc) / sd(xc)
> beta1c
[1] 0.6462906
> beta0c <- mean(yc) - beta1c * mean(xc)
> beta0c
[1] -1.50024e-16
> lm(yc ~ xc)

Call:
lm(formula = yc ~ xc)

Coefficients:
(Intercept)           xc
 -1.485e-16    6.463e-01

> lm(yc ~ xc - 1)

Call:
lm(formula = yc ~ xc - 1)

Coefficients:
    xc
0.6463
> plot(xc,yc)
> abline(coef = coef(lm(yc~xc)))
```

Since the line is defined by data that is centered, the resulting line goes through the the mean of x and y (0,0), we can calculate the slope by:

```
> sum(yc * xc) / sum(xc ^ 2)
[1] 0.6462906
```



Scaled Data

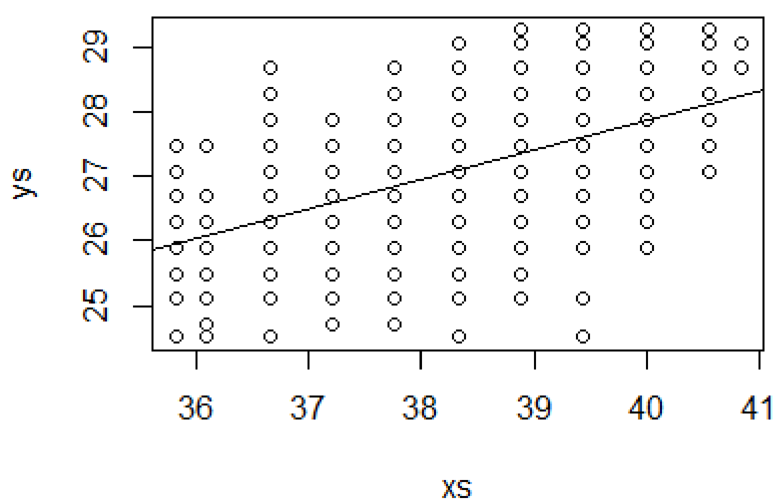
```
> ys <- y/sd(y)
> xs <- x/sd(x)
> beta1s <- cor(ys, xs) * sd(ys) / sd(xs)
> beta1s
[1] 0.4587624
> beta0s <- mean(ys) - beta1s * mean(xs)
> beta0s
[1] 9.508375
> lm(ys ~ xs)

Call:
lm(formula = ys ~ xs)

Coefficients:
(Intercept)           xs
     9.5084       0.4588
> lm(ys ~ xs - 1)

Call:
lm(formula = ys ~ xs - 1)

Coefficients:
    xs
0.7074
```

The data is scaled but not centered, If we want a line that goes through the origin (0,0), we can calculate the slope by:

```
> sum(ys * xs) / sum(xs ^ 2)
[1] 0.7073859
```

but it will return an error

```
> plot(xs,ys)
> abline(coef = coef(lm(ys~xs)))
```

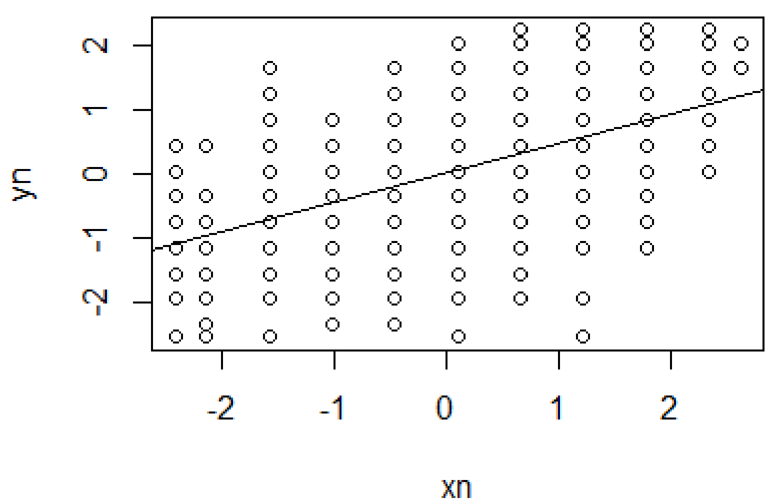


Normalized Data

```
> yn <- (y - mean(y))/sd(y)
> xn <- (x - mean(x))/sd(x)
> beta1n <- cor(yn, xn) * sd(yn) / sd(xn)
> beta1n
[1] 0.4587624
> beta0n <- mean(yn) - beta1n * mean(xn)
> beta0n
[1] -3.400448e-17
> lm(yn ~ xn)

Call:
lm(formula = yn ~ xn)

Coefficients:
(Intercept)           xn
 -3.545e-17    4.588e-01

> lm(yn ~ xn - 1)

Call:
lm(formula = yn ~ xn - 1)

Coefficients:
    xn
0.4588

> plot(xn,yn)
> abline(coef = coef(lm(yn~xn)))
```

The data is scaled and centered, the line goes through the mean of x and y (0,0), we can calculate the slope by:

```
> sum(yn * xn) / sum(xn ^ 2)
[1] 0.4587624
```



Comparing slope using lm

```
> coef(lm(y ~ x, galton))
(Intercept)           x
 23.9415302   0.6462906

> coef(lm(yc ~ xc))
  (Intercept)            xc
-1.485327e-16  6.462906e-01

> coef(lm(ys ~ xs))
(Intercept)          xs
  9.5083748   0.4587624

> coef(lm(yn ~ xn))
  (Intercept)            xn
-3.544656e-17  4.587624e-01
```

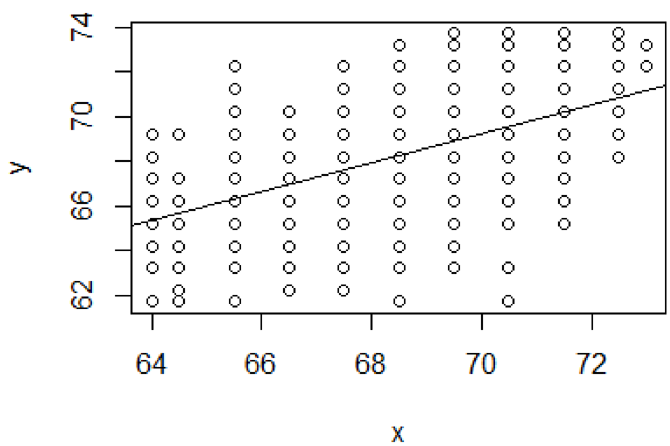
Comparing slope using sum(y * x) / sum(x ^ 2)

```
> sum(y * x) / sum(x ^ 2)
[1] 0.9965439

> sum(yc * xc) / sum(xc ^ 2)
[1] 0.6462906

> sum(ys * xs) / sum(xs ^ 2)
[1] 0.7073859

> sum(yn * xn) / sum(xn ^ 2)
[1] 0.4587624
```

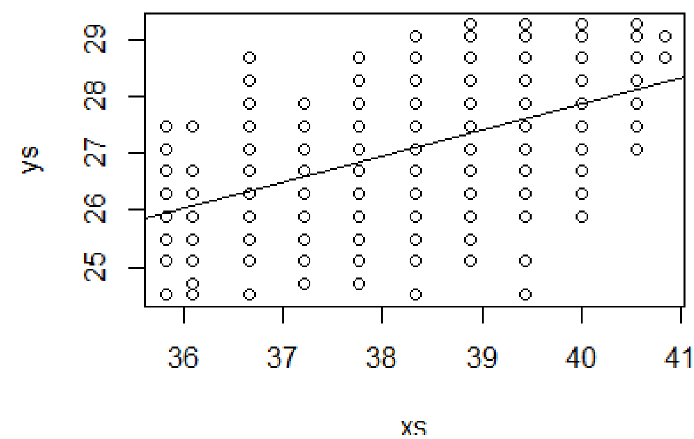
Regression through the origin

```
> coef(lm(y ~ x - 1, galton))
        x
0.9965439

> coef(lm(yc ~ xc - 1, galton))
       xc
0.6462906

> coef(lm(ys ~ xs - 1, galton))
       xs
0.7073859

> coef(lm(yn ~ xn - 1, galton))
       xn
0.4587624
```

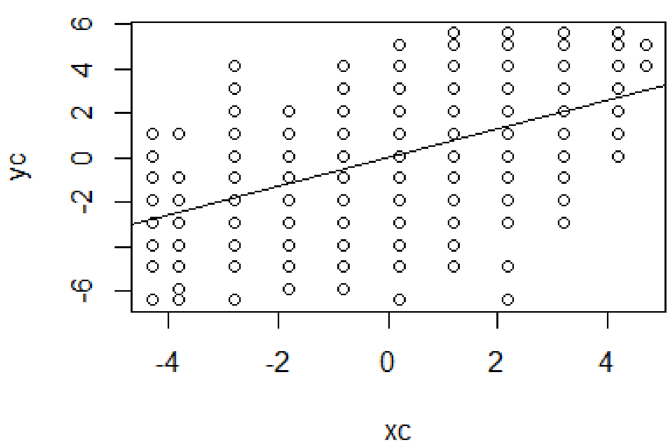
Comparing plots



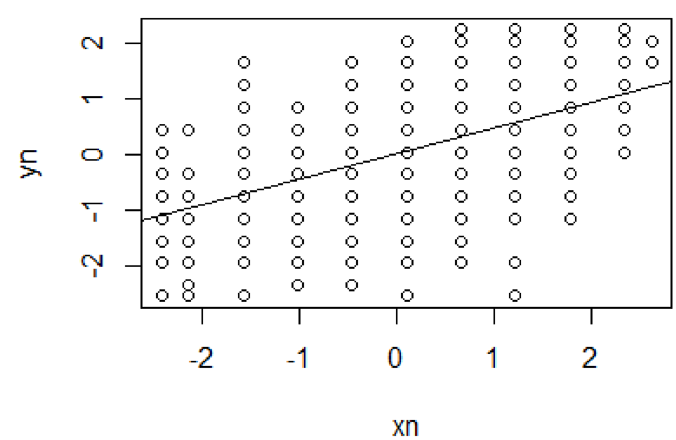

```
> coef(lm(y ~ x, galton))
(Intercept)           x
 23.9415302   0.6462906
```




```
> coef(lm(ys ~ xs))
(Intercept)          xs
  9.5083748   0.4587624
```




```
> coef(lm(yc ~ xc))
  (Intercept)            xc
-1.485327e-16  6.462906e-01
```




```
> coef(lm(yn ~ xn))
  (Intercept)            xn
-3.544656e-17  4.587624e-01
```