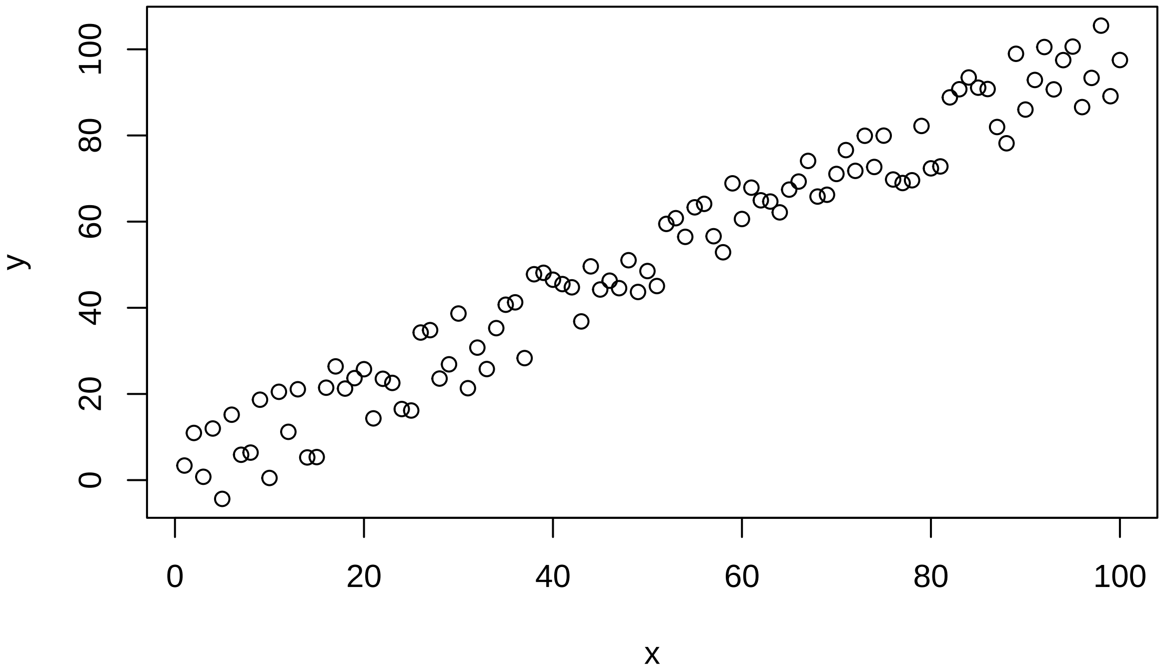
Interrupted Time Series

Let's create some data.

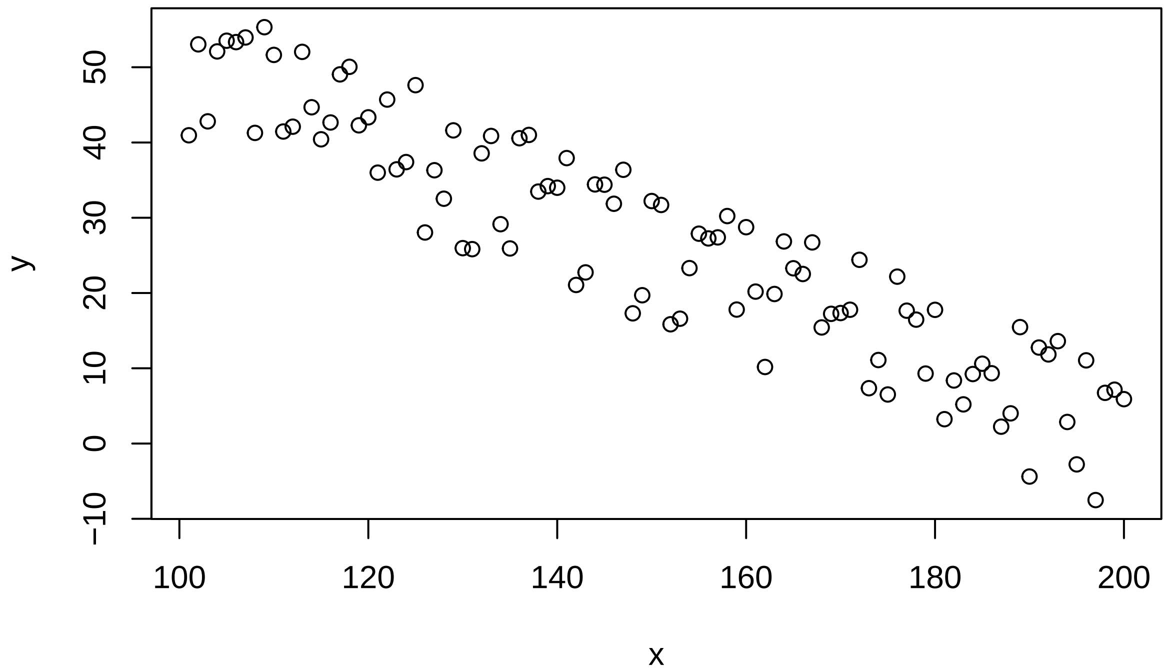
Here's a positive relationship.

```
j = 50

a = data.frame(x=1:100, y=jitter(1:100, j))
plot(a)
```



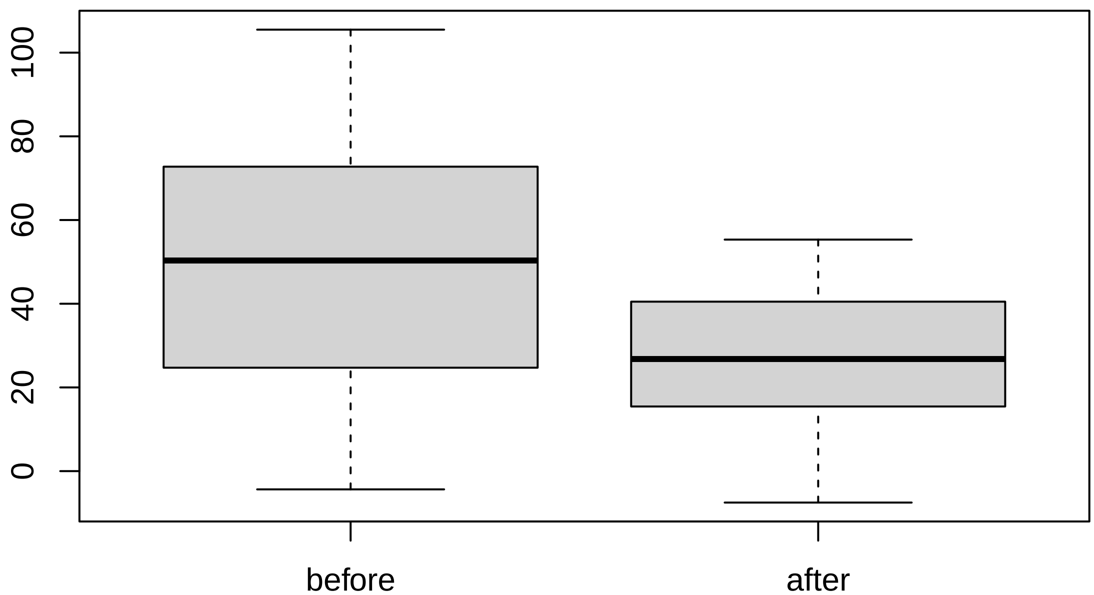
Here's a negative relationship.

```
b = data.frame(x=101:200, y=jitter(100:1, j))
bb = data.frame(x=101:200, y=jitter(seq(50,0.5,-0.5), 100))
plot(bb)
```



Are these any different?

```
boxplot(list(before=a$y,after=bb$y))
```



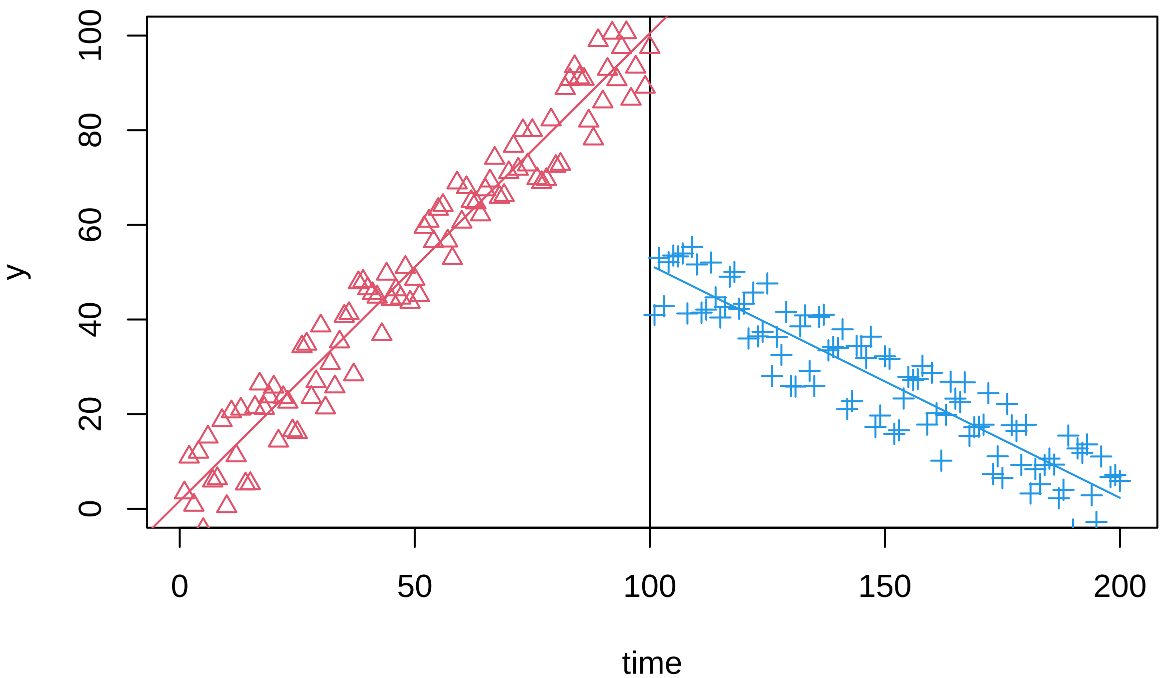

```
t.test(a$y,b$y)
```

```
##
##  Welch Two Sample t-test
##
## data:  a$y and b$y
## t = 0.18863, df = 197.99, p-value = 0.8506
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.417748  8.986885
## sample estimates:
```

```
## mean of x mean of y
##  51.56097  50.77640
```

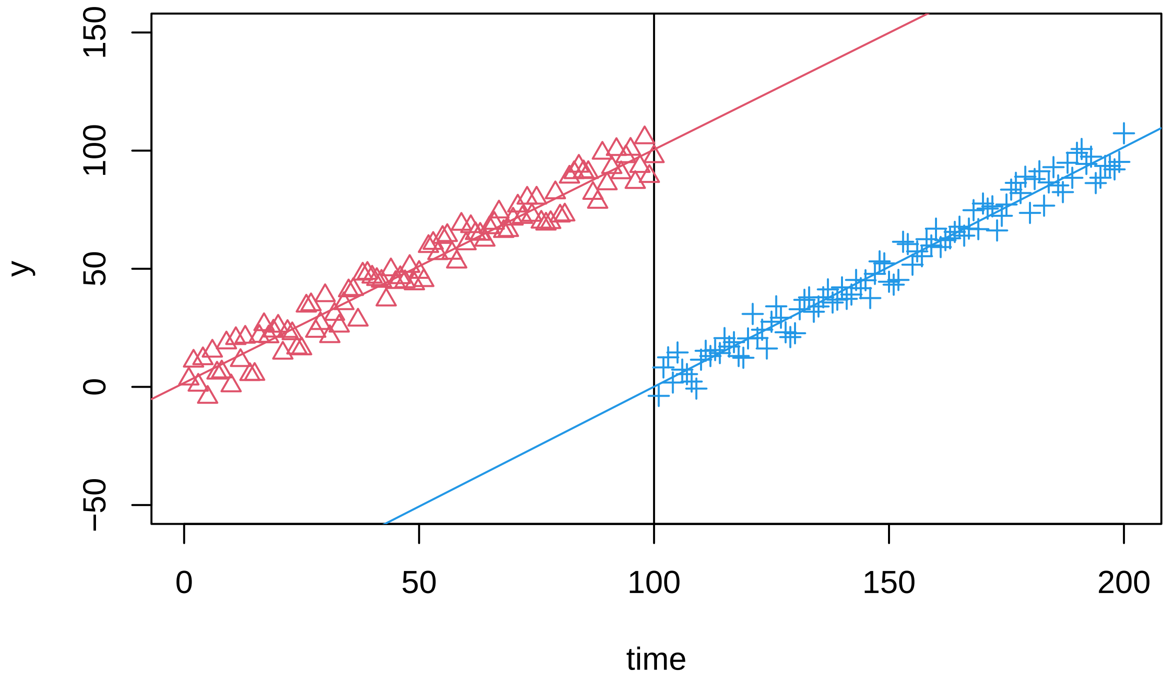
Let's display them side by side.

```r
plot(x=1:200, y=rep(1,200), type="n", ylim=c(0,100),
     xlab="time", ylab="y")
abline(v=100)
points(a$x, a$y, pch=2, col=2)
points(bb$x, bb$y, pch=3, col=4)

abline(lm(y~x, data=a), col=2)
lines(x=1:100, y=lm(y~x, data=a)$fit, col=2)
# abline(lm(y~x, data=bb), col=4)
lines(x=101:200, y=lm(y~x, data=bb)$fit, col=4)
```



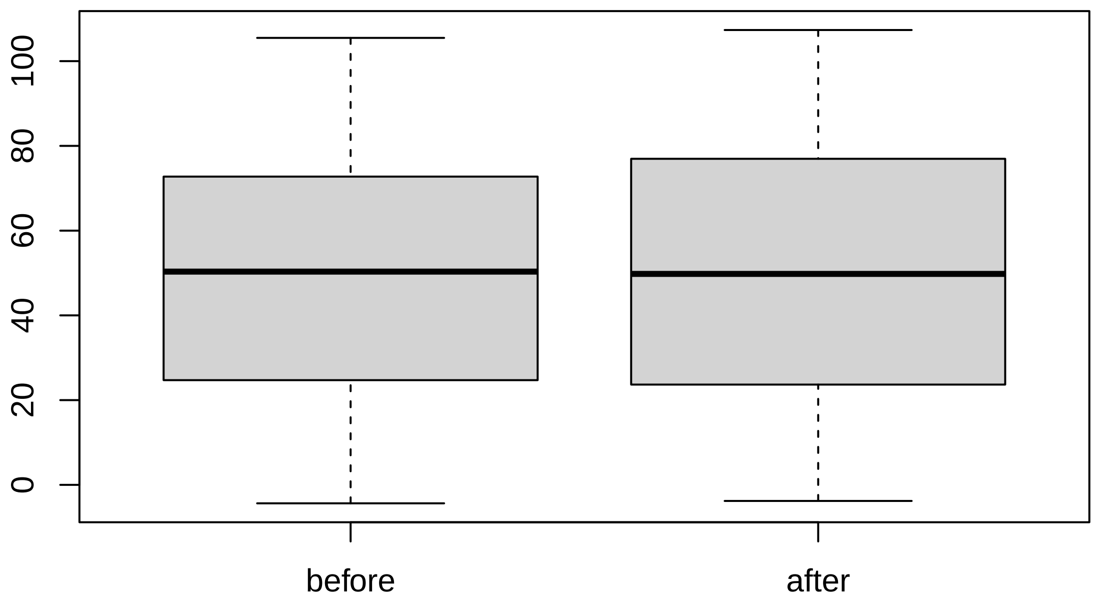
Let's simulate a change in level.

```r
a2 = data.frame(x=101:200, y=jitter(1:100, j))

plot(x=1:200, y=rep(1,200), type="n", ylim=c(-50,150),
     xlab="time", ylab="y")
abline(v=100)
points(a$x, a$y, pch=2, col=2)
points(a2$x, a2$y, pch=3, col=4)

abline(lm(y~x, data=a), col=2)
abline(lm(y~x, data=a2), col=4)
```



We can't capture that with a simple test.

```
boxplot(list(before=a$y,after=a2$y))
```



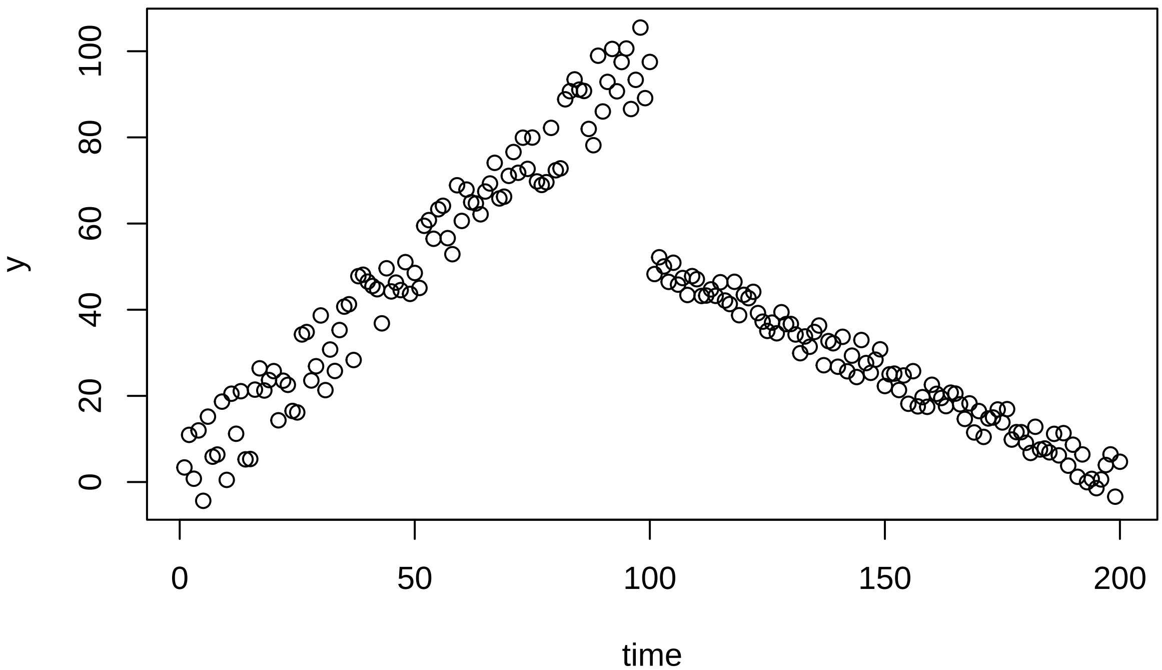

```
t.test(a$y,a2$y)
```

```
##
##  Welch Two Sample t-test
##
## data:  a$y and a2$y
## t = 0.062368, df = 197.93, p-value = 0.9503
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.992779  8.514855
```

```
## sample estimates:
## mean of x mean of y
##  51.56097  51.29993
```

Now let's go back to the previous example:

```r
m = rbind(a, data.frame(x=101:200, y=jitter(seq(50,0.5,-0.5), j)))
plot(m$x, m$y, xlab="time", ylab="y")
```



Here's what a simple model might look like:

```r
summary(lm(y~x, data=m))
```

```
##
## Call:
## lm(formula = y ~ x, data = m)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -55.846 -17.459  -1.771  11.772  66.756
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 52.17421    3.62637  14.387  < 2e-16 ***
## x           -0.13715    0.03129  -4.383 1.89e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.55 on 198 degrees of freedom
## Multiple R-squared:  0.08846,    Adjusted R-squared:  0.08385
## F-statistic: 19.21 on 1 and 198 DF,  p-value: 1.895e-05
```

Let's see if we can model those trends and change in level explicitly.

```r
m$time = m$x
m$intervention = m$time > 100
m$time_after_intervention = ifelse(m$time > 100, m$time - 100, 0)

m$time
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
## [127] 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
## [145] 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
## [163] 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
## [181] 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
## [199] 199 200
```

```r
m$intervention
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [109]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [145]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [157]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [169]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [181]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [193]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
m$time_after_intervention
```

```
##   [1]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [19]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [37]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [55]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [73]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [91]   0   0   0   0   0   0   0   0   0   0   1   2   3   4   5   6   7   8
## [109]   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26
## [127]  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44
## [145]  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62
## [163]  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
## [181]  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98
## [199]  99 100
```

```r
rdd = lm(y ~ time + intervention + time_after_intervention, data=m)
summary(rdd)
```

```
##
## Call:
## lm(formula = y ~ time + intervention + time_after_intervention,
##     data = m)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1442  -2.9659   0.2115   3.4002   9.3657
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              1.67710    0.97135   1.727   0.0858 .
## time                     0.98780    0.01670  59.153   <2e-16 ***
## interventionTRUE       -49.75945    1.36350 -36.494   <2e-16 ***
## time_after_intervention  -1.49229    0.02362 -63.190   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.82 on 196 degrees of freedom
## Multiple R-squared:  0.9679, Adjusted R-squared:  0.9674
## F-statistic:  1968 on 3 and 196 DF,  p-value: < 2.2e-16
```

Q: Can you achieve the same result (i.e., capture both trends and the change in level) with only two variables?
A: Yes, with an interaction term!

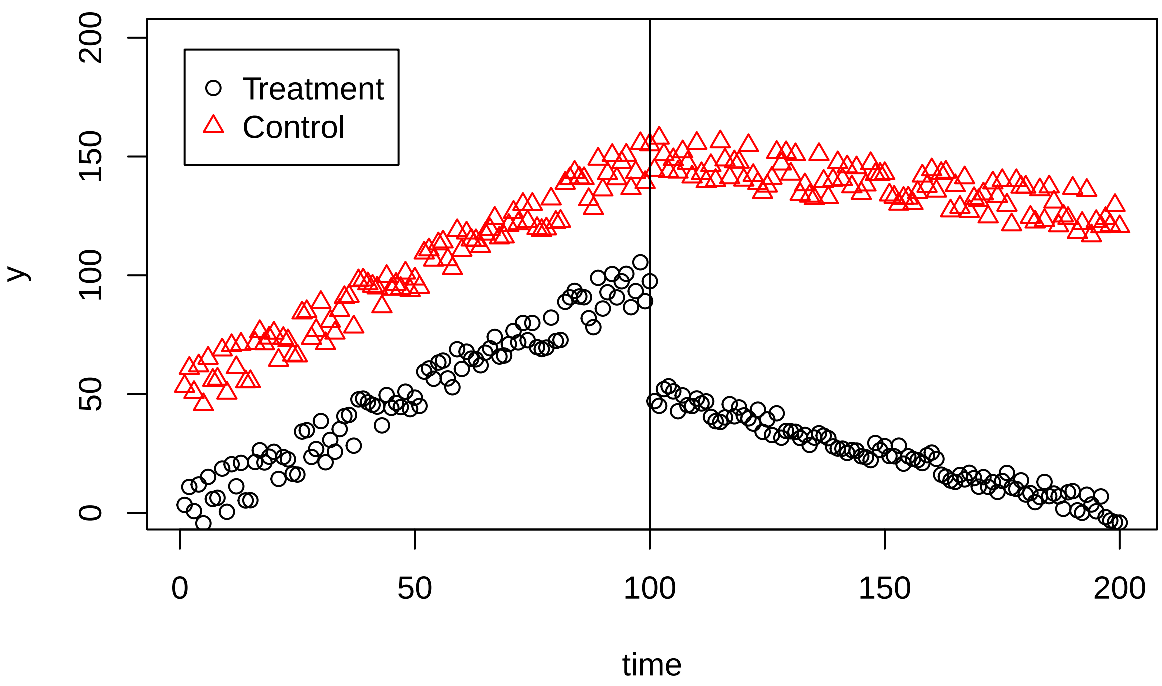
```r
rdd2 = lm(y ~ time * intervention, data=m)
summary(rdd2)
```

```
##
## Call:
## lm(formula = y ~ time * intervention, data = m)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1442  -2.9659   0.2115   3.4002   9.3657
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.67710    0.97135   1.727   0.0858 .
## time                   0.98780    0.01670  59.153   <2e-16 ***
## interventionTRUE      99.46970    2.73716  36.340   <2e-16 ***
## time:interventionTRUE -1.49229    0.02362 -63.190   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.82 on 196 degrees of freedom
## Multiple R-squared:  0.9679, Adjusted R-squared:  0.9674
## F-statistic:  1968 on 3 and 196 DF,  p-value: < 2.2e-16
```

Now let's add a control series.

```r
a2 = a
names(a2) = c("x","yt")
df = rbind(a2, data.frame(x=101:200, yt=jitter(seq(50,0.5,-0.5), j)))
df$yc = jitter(50) + df$yt
df[df$x>=100,]$yc = jitter(seq(150,125,-0.25), 4*j)
```

```
{
  plot(df$x, type="n", xlab="time", ylab="y")
  points(df$x, df$yt)
  points(df$x, df$yc, col = "red", pch=2)
  legend(1, 195, legend=c("Treatment", "Control"),
       col=c("black", "red"), pch=c(21,2))
  abline(v=100)
}
```



And set up the ITS variables.

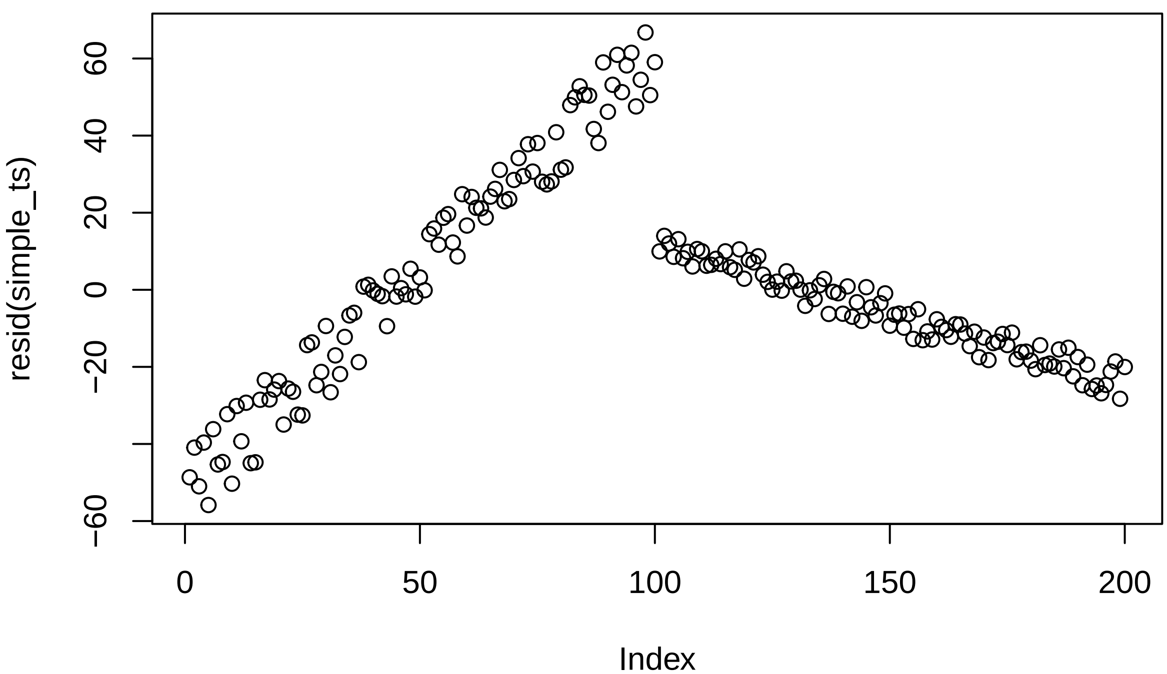
```
dfm = data.frame(time = df$x, y = c(df[c("x","yt")]$yt,df[c("x","yc")]$yc))

dfm$group = c(rep("treated",200), rep("control",200))
dfm$intervention = dfm$time > 100
dfm$time_after_intervention = ifelse(dfm$time > 100, dfm$time - 100, 0)

rdd2c = lm(y ~ time
           + intervention
           + time_after_intervention
           + group
           + group:time
           + group:intervention
           + group:time_after_intervention
           , data=dfm)
summary(rdd2c)
```

```
##
## Call:
## lm(formula = y ~ time + intervention + time_after_intervention +
##     group + group:time + group:intervention + group:time_after_intervention,
##     data = dfm)
##
```

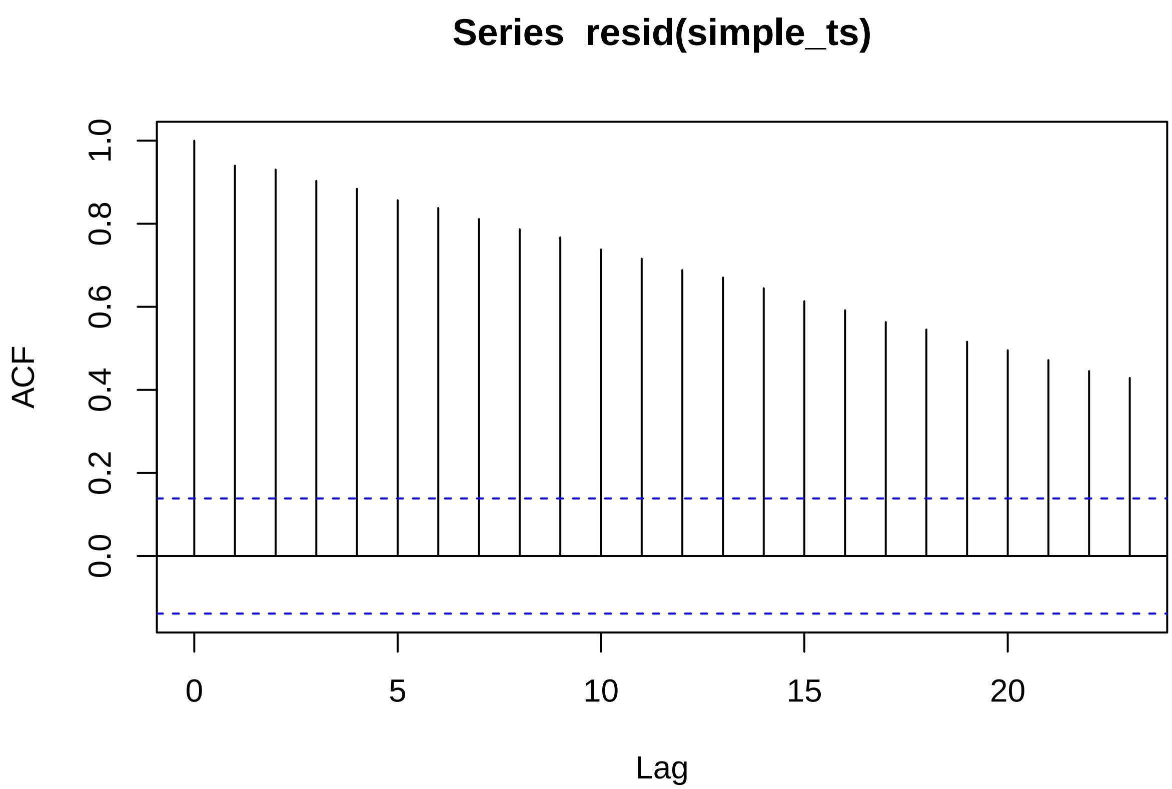
```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1442  -3.7472  -0.1694   4.4578   9.9158
##
## Coefficients:
##                                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          51.481653   1.122071  45.881  < 2e-16 ***
## time                                  0.992247   0.019290  51.438  < 2e-16 ***
## interventionTRUE                     -0.638389   1.575079  -0.405    0.685
## time_after_intervention              -1.243818   0.027280 -45.594  < 2e-16 ***
## grouptreated                        -49.804550   1.586847 -31.386  < 2e-16 ***
## time:grouptreated                    -0.004448   0.027280  -0.163    0.871
## interventionTRUE:grouptreated       -49.764833   2.227498 -22.341  < 2e-16 ***
## time_after_intervention:grouptreated  -0.253254   0.038580  -6.564 1.66e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.568 on 392 degrees of freedom
## Multiple R-squared:  0.9874, Adjusted R-squared:  0.9872
## F-statistic:  4404 on 7 and 392 DF,  p-value: < 2.2e-16
```

Is there autocorrelation?

```r
simple_ts = lm(y ~ time, data=m)
plot(resid(simple_ts))
```




```r
# alternatively
acf(resid(simple_ts))
```



To formally test for autocorrelation, we can use the Durbin-Watson test

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.4.1

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
dwtest(m$y ~ m$time)
```

```
##
##  Durbin-Watson test
##
## data:  m$y ~ m$time
## DW = 0.098841, p-value < 2.2e-16
## alternative hypothesis: true autocorrelation is greater than 0
```

From the p-value, we know that there is autocorrelation in the time series

A solution to this problem could be to use more advanced time series analysis (e.g., ARIMA) to adjust for seasonality and other dependency, or to use mixed-effects models when modeling multiple individual “treated” time series jointly.