Regression with R

Data

For a certain algorithm the time to calculate the results is measured for different input sizes and recorded in a text file.

Read this data into R using the read.table function.

```
# Read the data
data = read.table("data.dat", sep='', header=FALSE)
data
```

```
##      V1    V2
## 1   280 0.015
## 2   316 0.016
## 3   494 0.016
## 4  1347 0.031
## 5  1463 0.031
## 6  2872 0.063
## 7  3302 0.094
## 8  3717 0.094
## 9  4711 0.125
## 10 5408 0.140
## 11 6410 0.156
## 12 6417 0.156
## 13 6656 0.172
## 14 7251 0.187
## 15 7294 0.188
## 16 7879 0.204
## 17 7883 0.203
## 18 8097 0.187
## 19 9684 0.250
## 20 9901 0.250
```

Data analysis: log10 transformation

Let's see how well the data fits a model after a log10 transform.

```
logn = log10(data$V1)
mlogn = lm(data$V2 ~ logn)
summary(mlogn)
```

```
##
## Call:
## lm(formula = data$V2 ~ logn)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.054466 -0.025922  0.002619  0.021870  0.055585
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.38665    0.05693  -6.792 2.32e-06 ***
## logn         0.14577    0.01595   9.139 3.50e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03418 on 18 degrees of freedom
## Multiple R-squared:  0.8227, Adjusted R-squared:  0.8128
## F-statistic: 83.52 on 1 and 18 DF,  p-value: 3.504e-08
```

Data analysis: square transformation

Let's see how well the data fits a model after a n^2 transform.

```r
nsq = data$V1^2
mnsq = lm(data$V2 ~ nsq)
summary(mnsq)
```

```
##
## Call:
## lm(formula = data$V2 ~ nsq)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.032830 -0.018465  0.008582  0.015881  0.027777
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.291e-02  7.371e-03   5.821 1.63e-05 ***
## nsq         2.448e-09  1.588e-10  15.413 8.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02154 on 18 degrees of freedom
## Multiple R-squared:  0.9296, Adjusted R-squared:  0.9257
## F-statistic: 237.6 on 1 and 18 DF,  p-value: 8.178e-12
```

Data analysis: n*log10 transformation

Let's see how well the data fits a model after a n*log10(n) transform.

```r
nlogn = data$V1 * log10(data$V1)
mnlogn = lm(data$V2 ~ nlogn)
summary(mnlogn)
```

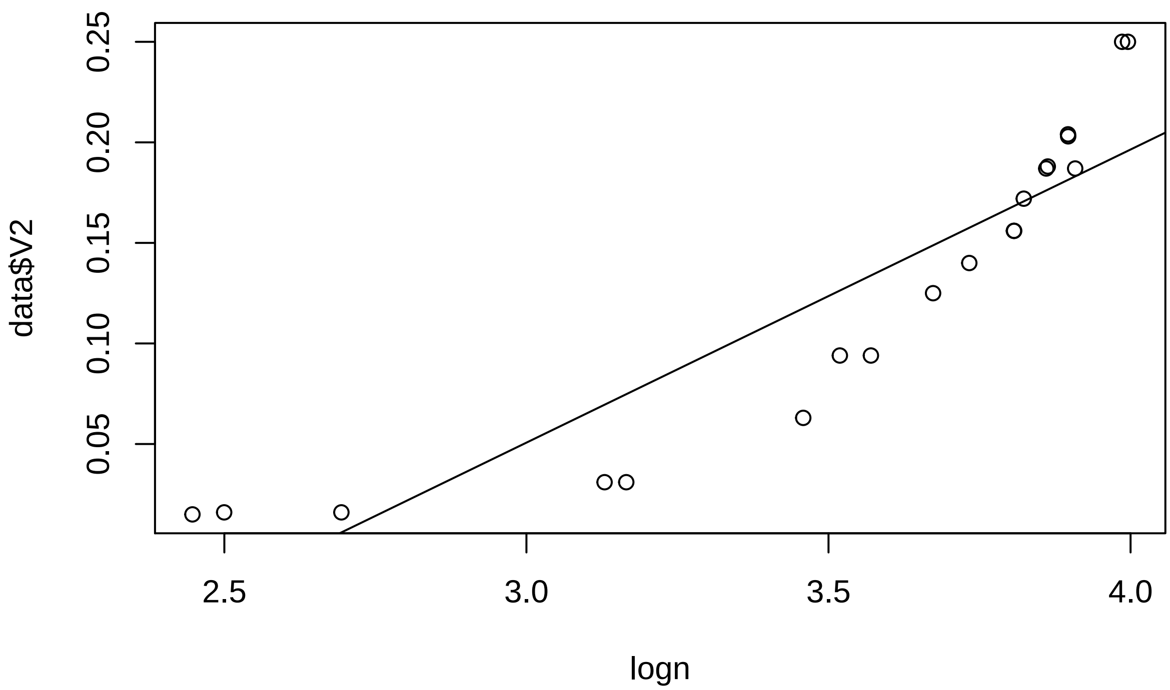
```
##
## Call:
## lm(formula = data$V2 ~ nlogn)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.018288 -0.004579  0.001546  0.004354  0.012445
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.768e-03  2.922e-03   3.343  0.00362 **
## nlogn       6.178e-06  1.274e-07  48.487  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007075 on 18 degrees of freedom
## Multiple R-squared:  0.9924, Adjusted R-squared:  0.992
## F-statistic:  2351 on 1 and 18 DF,  p-value: < 2.2e-16
```

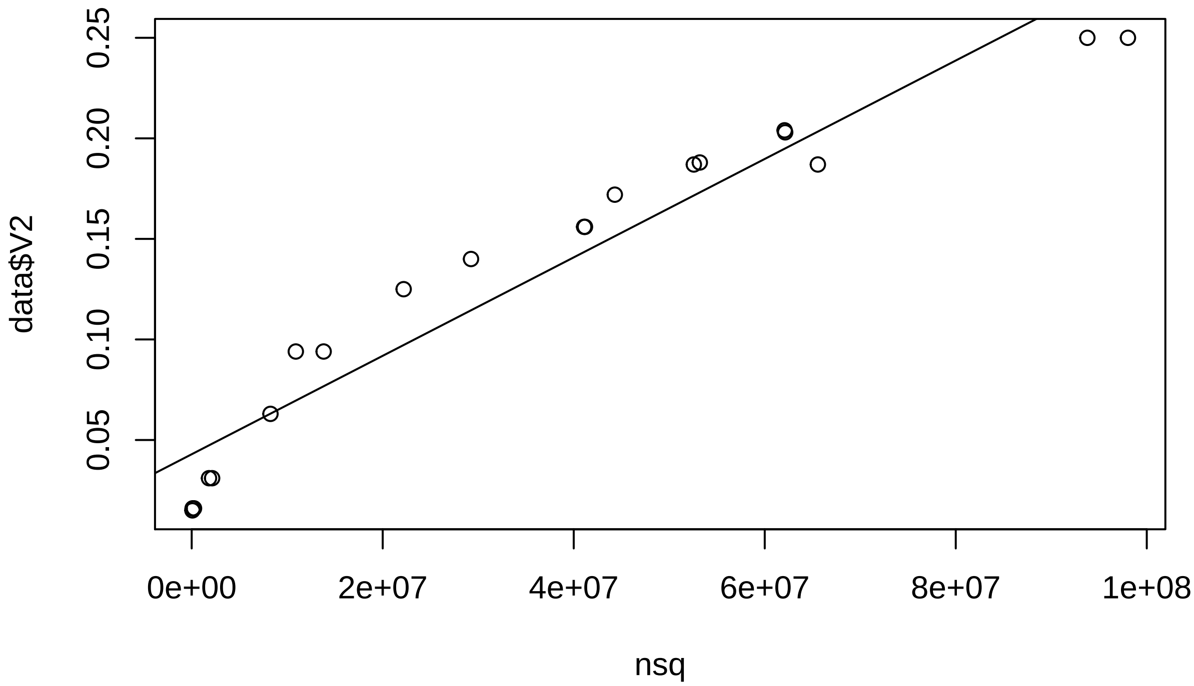
Visual analysis

Create graphs for each combination. Plot the regression line.

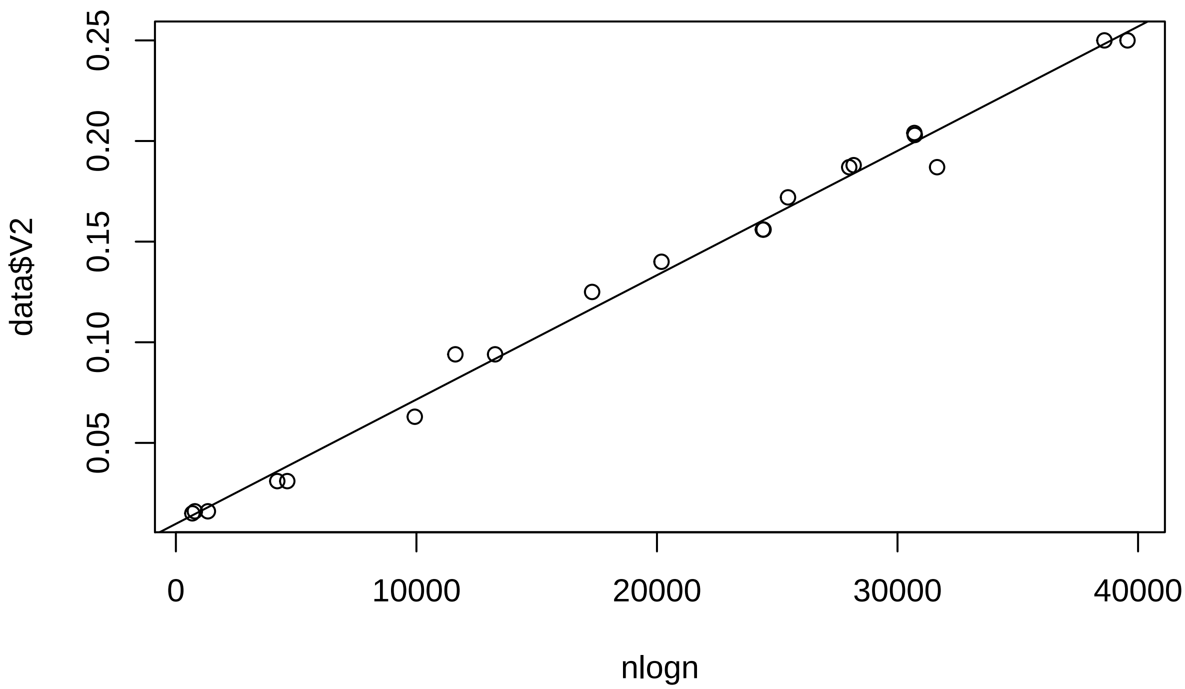
```
plot(logn,data$V2)
abline(mlogn)
```




```
plot(nsq,data$V2)
abline(mnsq)
```




```
plot(nlogn,data$V2)
abline(mnlogn)
```



Discuss the results

The R-square (R^2) value indicates the fit. Find the model where R^2 is the largest. The closer it is to 1, the better the fit. In this example n*log(n) has the best fit. Therefor we can conclude that the algorithm likely has an order of O(n log n).