Exploratory data analysis

Dominik Klepl

11/26/2019

In this project, we are working with 2 EEG signals. The goal of the project is model signal y as dependent variable predicted from signal x. We start with an exploratory analysis to better understand the nature of both signals. We look at distributions of both signals and test whether they follow normal distribution. We'll also investigate possible relationships between the two variables both by eyeballing scatterplots and by a statistical test of correlation. Finally, we'll fit a simple linear model y ~ b1*x + error, generate predictions and their uncertainty, do a quick residual analysis. As little bonus we use local polynomial regression fitting to see if we can get some hints about the underlaying true model that generated the y signal.

Load the dataset

```
data = read.csv("data/x_y.csv", header = F)
```

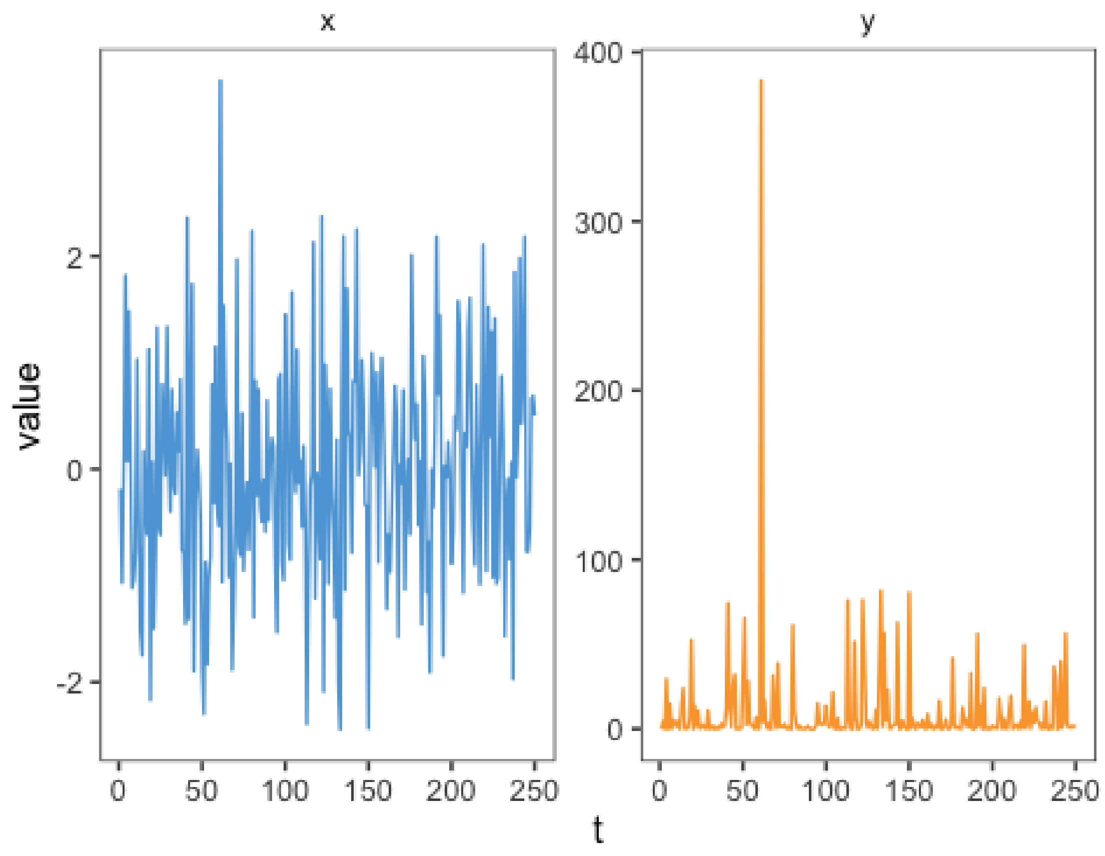
Rename the column names to x and y

```
colnames(data) = c("x", "y")
```

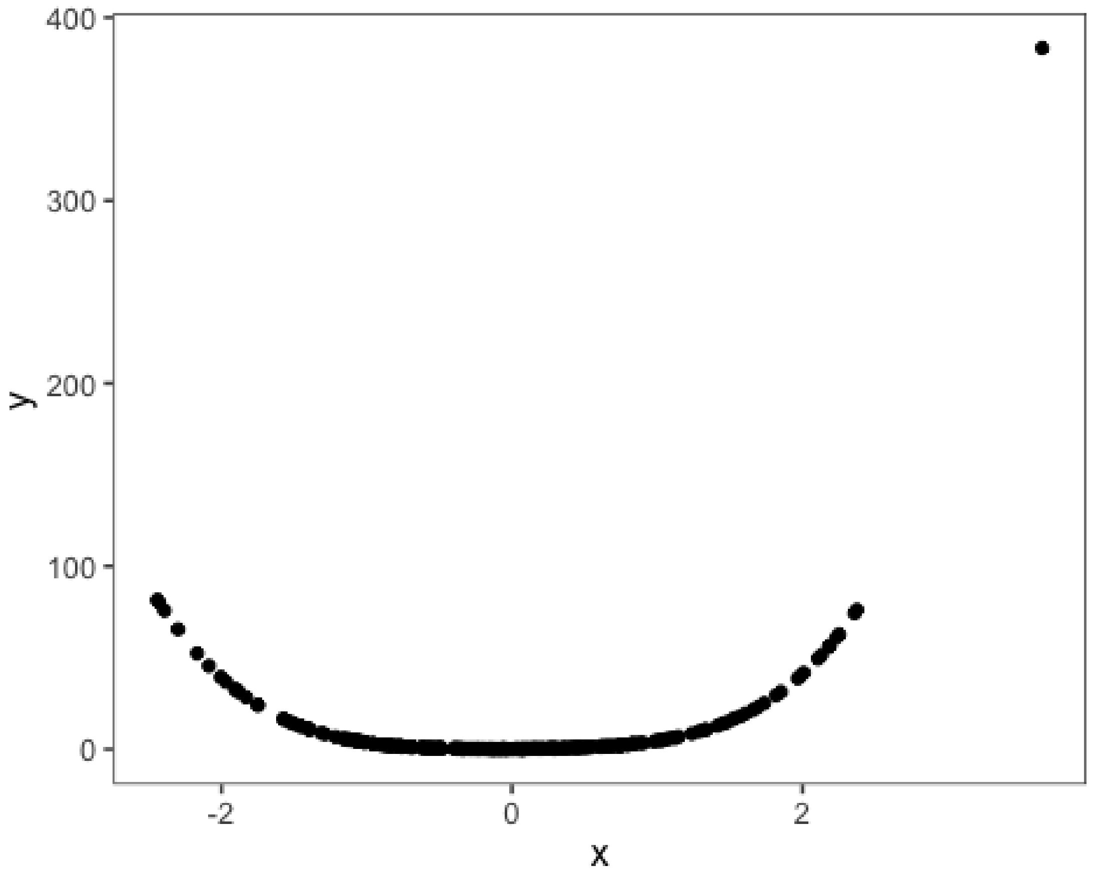
Add time variable to preserve the time-series structure

Relationship between x and y

We start with inspecting the input/output variables by plotting them. First on the same axis simply as two time-series signals.

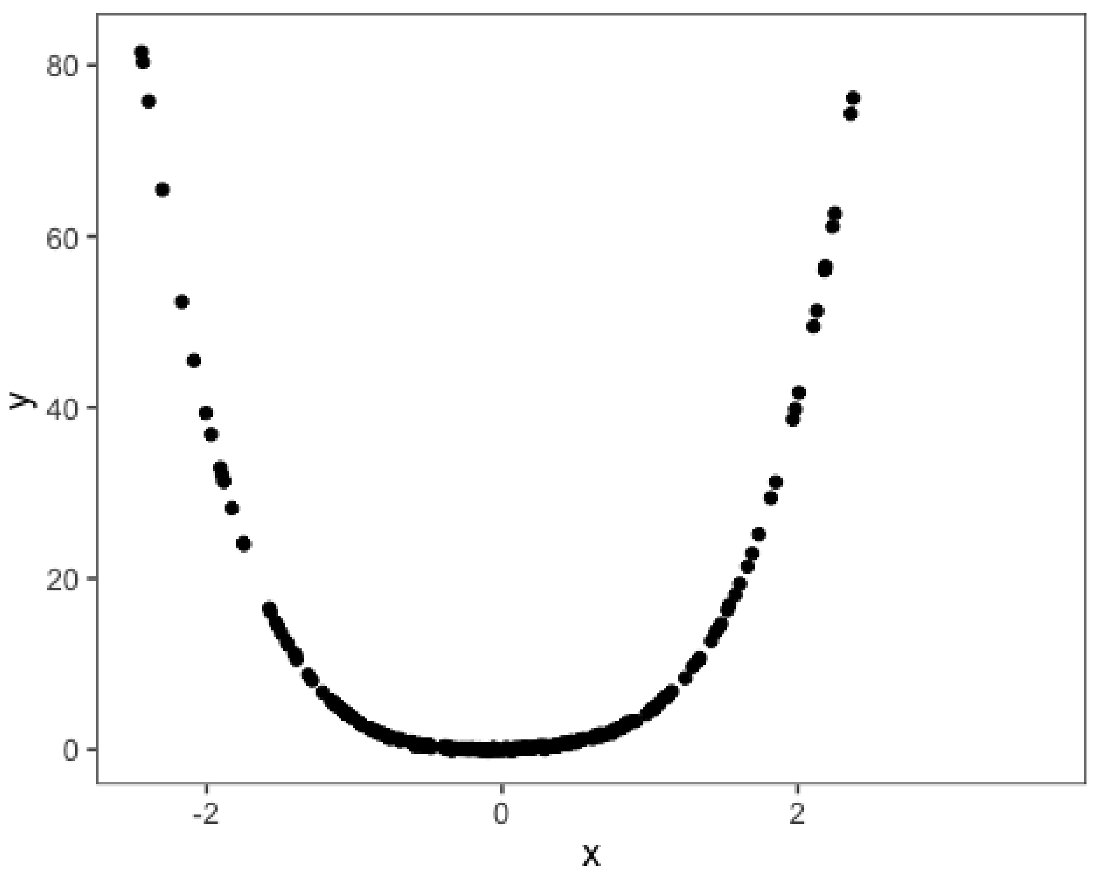


Now we also plot the signals against each other.



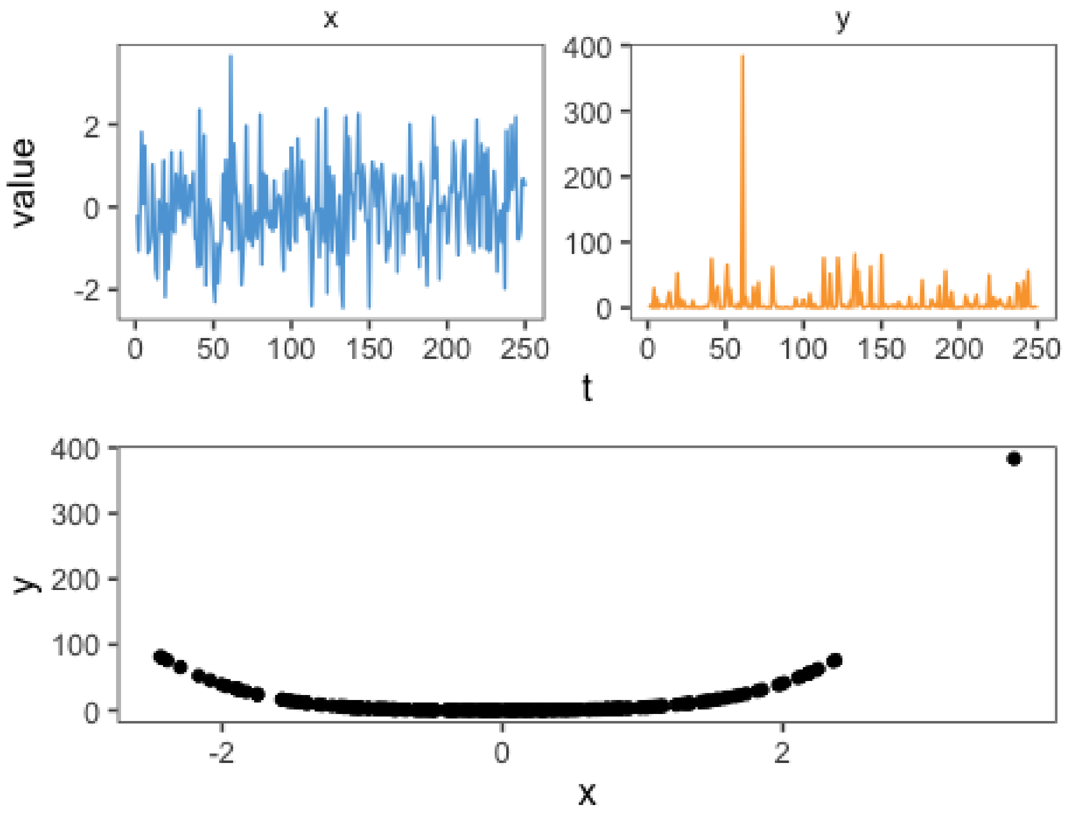
One point seems to be an **outlier**.

It might be a good idea to to remove the outlier now for plotting so that we have a more detailed (zoomed-in) look at the rest of the datapoints.



The x^2 component is even clearer in the zoomed-in view.

Plot p1 and x_y_plot together in one *beautiful* plot.



From the scatterplot of the x and y variables we can assume that the a x^2 might be a good parameter for the model.

Correlation test

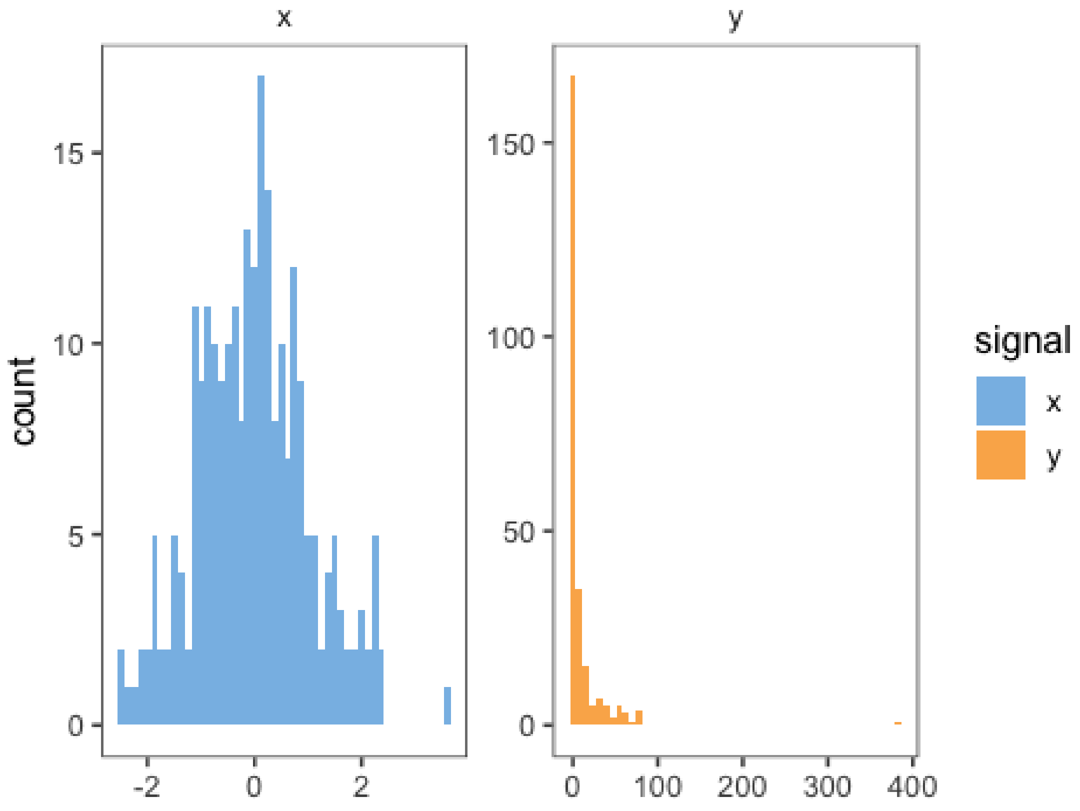
We can formally test whether there is correlation between x and y. Although we can already tell from the scatterplot that there must be some correlation. We can use **pearson's correlation coefficient**, testing hypothesis that true correlation differs from 0.

```
##
##  Pearson's product-moment correlation
##
## data:  data$x and data$y
## t = 3.5408, df = 248, p-value = 0.0004763
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.09796534 0.33433385
## sample estimates:
##       cor
## 0.2193661
```

There is small positive correlation between the two variables. Null hypothesis was rejected.

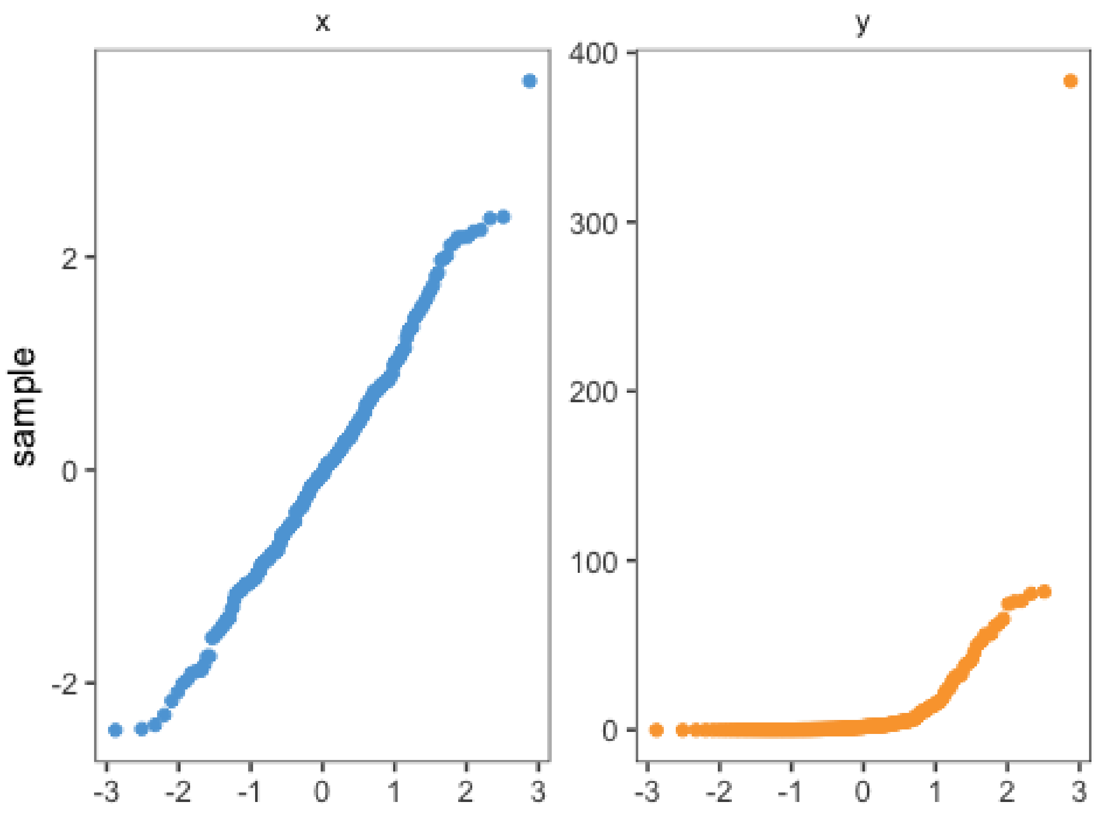
Distributions

Now we inspect the distribution of both x and y

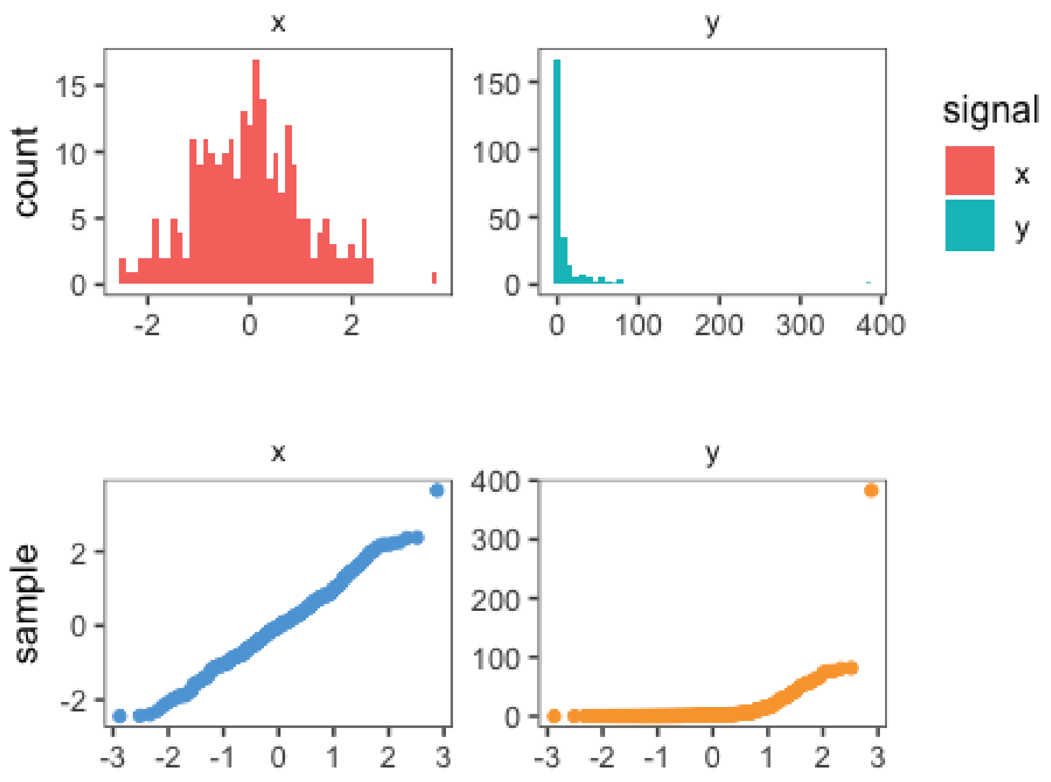


X seems to be approximately **normal** slightly skewed with heavy left tail. Y seems to be **exponentially distributed**. A hypothesis that y is **log-normal** might be worth testing.

QQ-plots



Combine histograms and qqplots into one beautiful plot



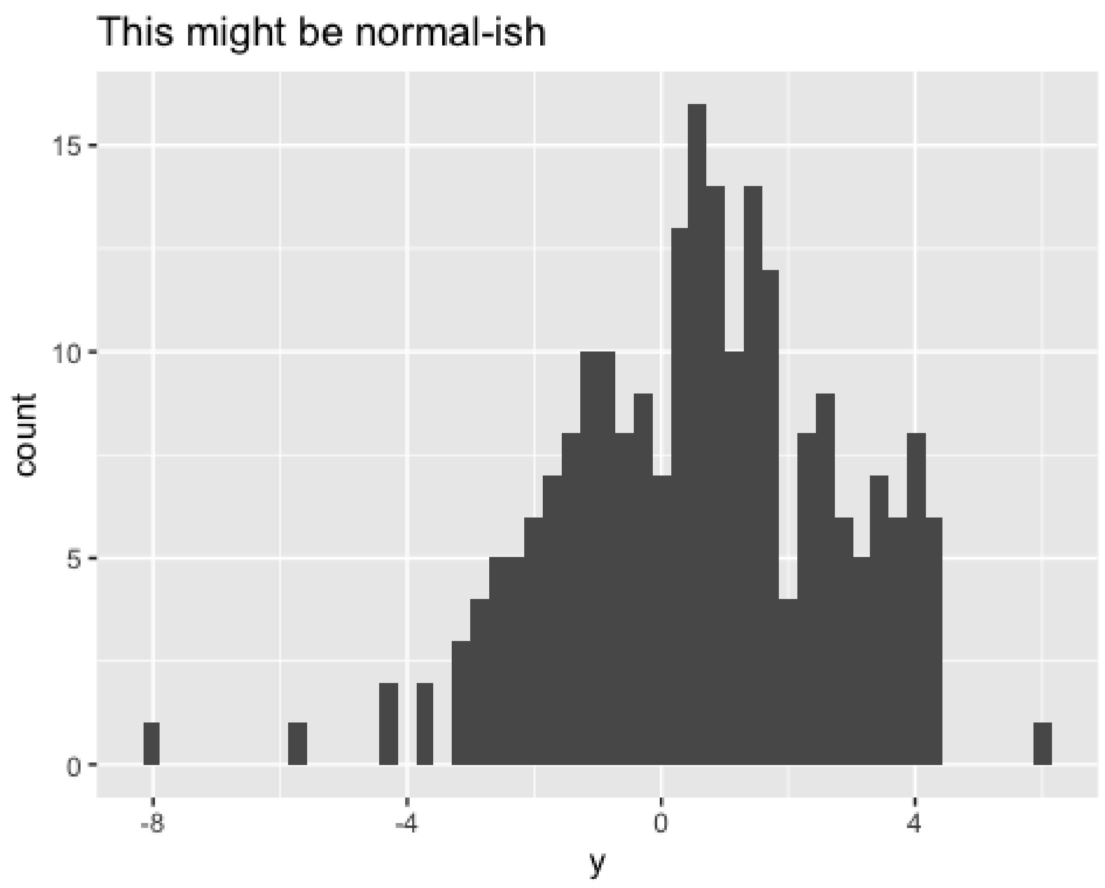
Further tests of normality

We can use Shapiro-Wilk test which tests the hypothesis whether variable is normally distributed. (Hoping for p-value < .05)

```
## Testing signal x

##
##  Shapiro-Wilk normality test
##
## data:  data$x
## W = 0.99297, p-value = 0.2854

## Testing signal y

##
##  Shapiro-Wilk normality test
##
## data:  data$y
## W = 0.32323, p-value < 2.2e-16
```

Just for fun - Is y log-normal?

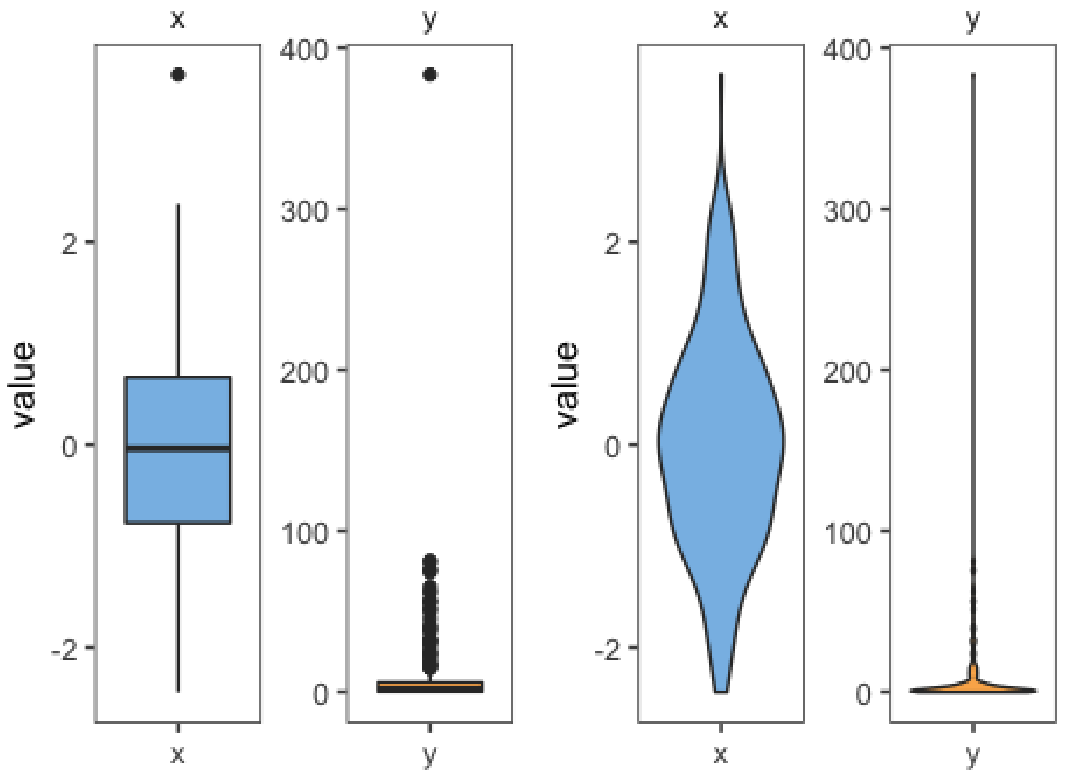
```
## Warning in log(data$y): NaNs produced
```



```
## How normal is log(y)?

##
##  Shapiro-Wilk normality test
##
## data:  log_norm$y
## W = 0.98585, p-value = 0.02352

## Wow, apparently more than anything else in this dataset
```

Boxplots and violin plots

Let’s continue with other tests about properties of the signals. First use boxplot and violin plots.



Fit linear model

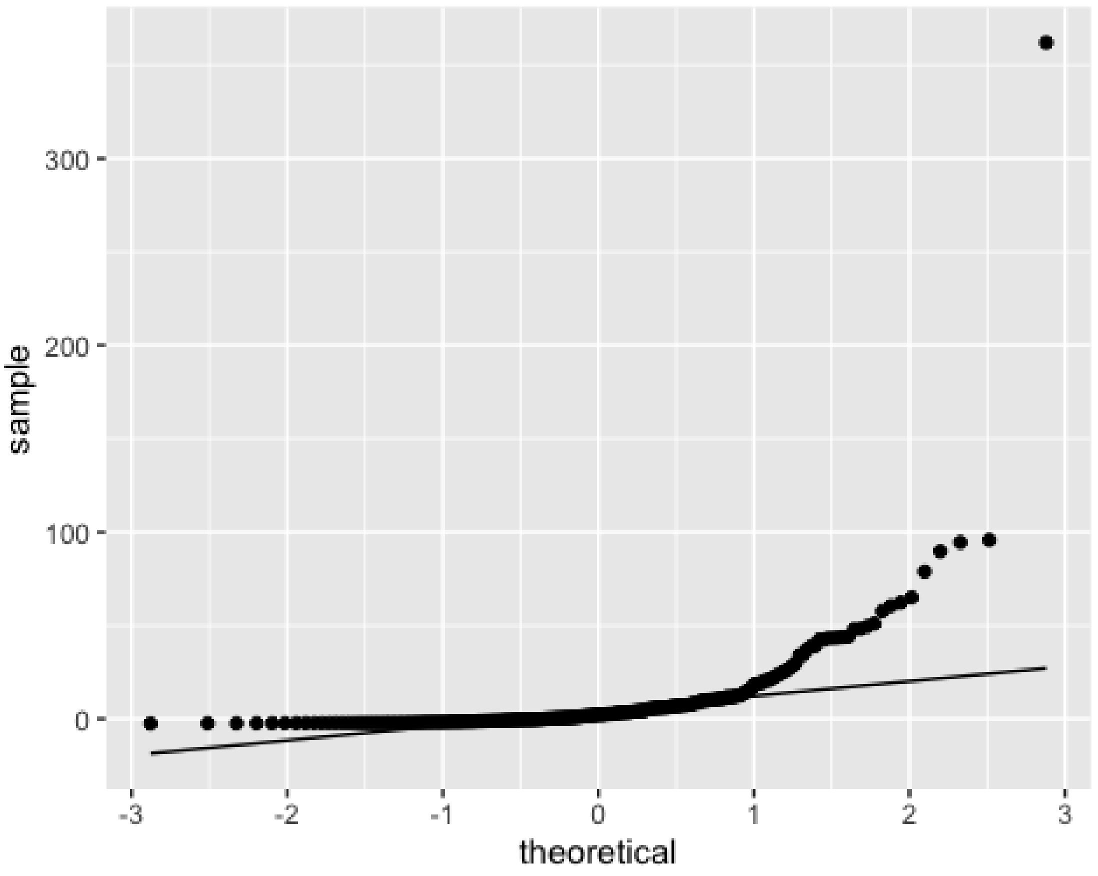
Try to fit a linear model with just one parameter: y ~ ß1*x

```
## MSE of the fitted model is: 886.242
```

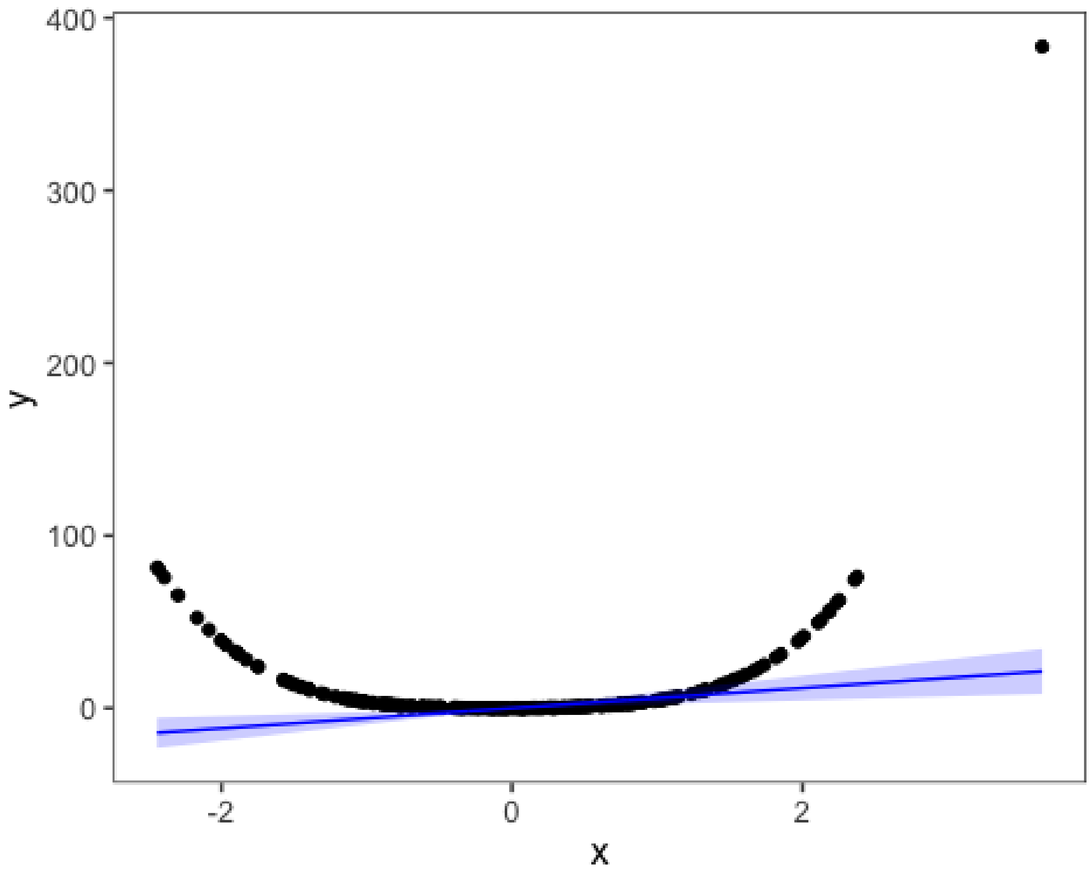
Residual analysis

```
## Residual analysis:

##         Min         25%      Median         75%         Max
##   -2.494014   -1.002830   2.178198   9.676968 361.995873
```

Plot residuals and qqplot

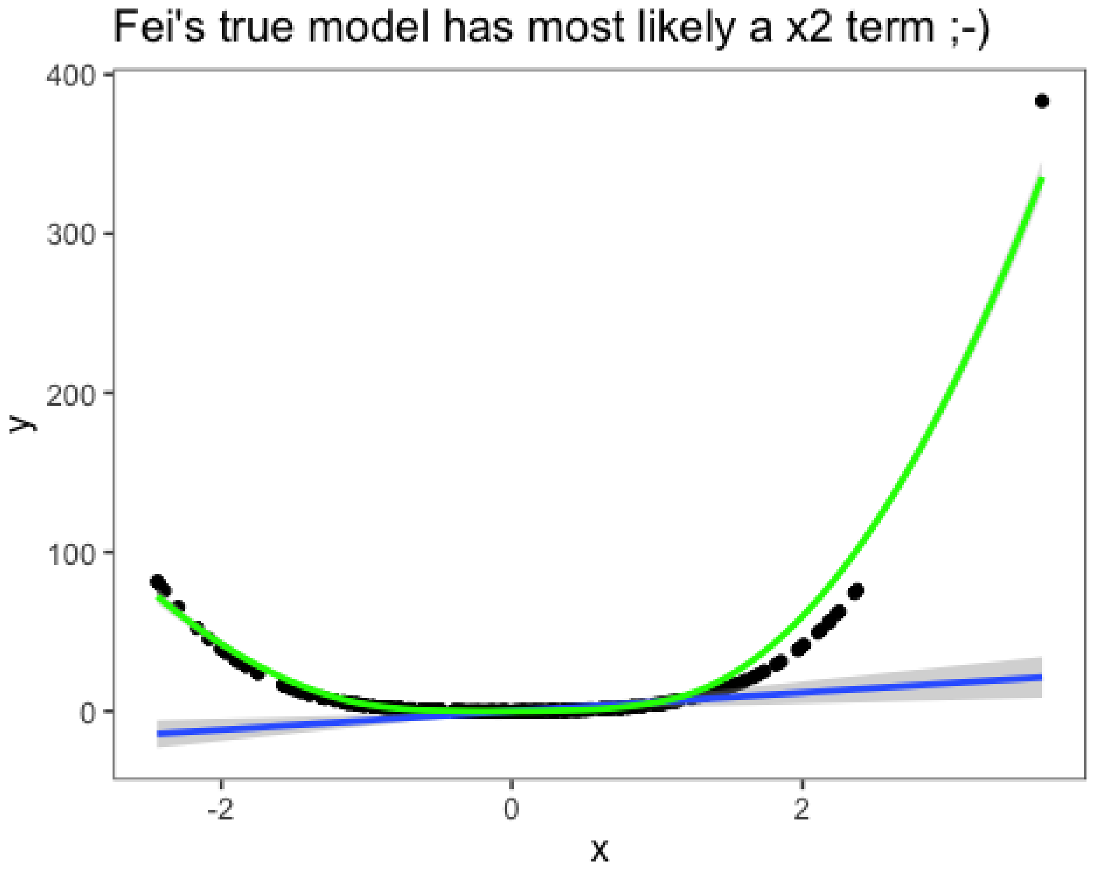


Plot model's predictions + 95% confidence intervals



A bit of "cheating"

Just for fun, ggplot has function for fitting a simple linear model (including confidence intervals). There's also function for fitting a local polynomial surface/line which basically tries to find the best polynomial model (yes exactly what is our task in the coursework).


Fei's true model has most likely a x2 term ;-)
y
x