Pre-Processing NO2 Data

Contents

Introduction

In this notebook investigates pre-processing steps which can be applied to NO2 concentration levels in oder to remove seasonal trends and a in 2018 for the four largest cities in Europe - London, Berlin, Madrid, and Rome.

```
##              London   Berlin   Madrid     Rome
## 2018-01-01 29.43041 34.08371 23.21701 42.90580
## 2018-01-02 38.77551 36.74439 33.28502 47.44203
## 2018-01-03 23.70369 29.93306 38.74034 59.88406
## 2018-01-04 37.29889 30.31549 31.21354 67.01087
## 2018-01-05 49.63385 39.17014 29.28684 56.64476
## 2018-01-06 49.00552 46.93307 26.10590 46.58333
```

Summary of pre-processing steps

Overall the following pre-processing steps will be applied to the cleaned data:

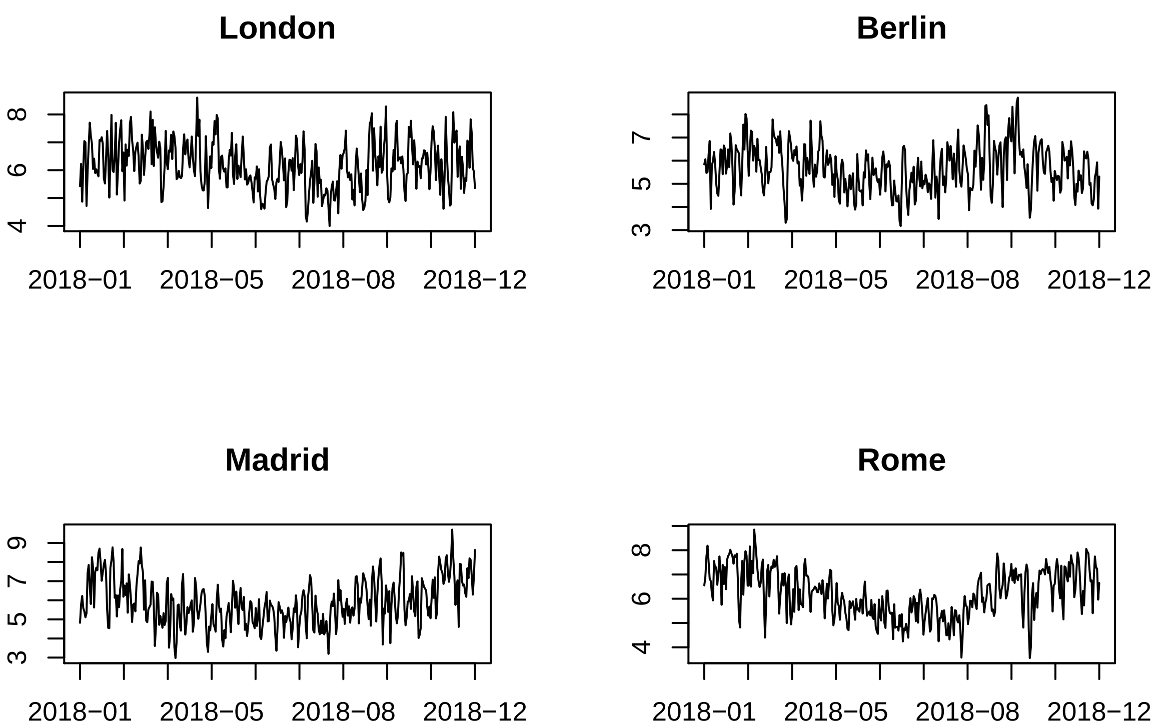
1. Square-root transform to bring the data back to the light tailed domain
2. Regression onto Fourier frequencies and indicators for weekends to eliminate trends; specifically frequencies corresponding to yearly and quarterly cycles will be used
3. Whitening of de-trended data by saving residuals from the fit of an AR model

In practice the linear model in step 2 and 3 should be applied to training data which does not contain change points, and the prediction error on the data being examined for change points may be used in placed or empirical residuals.

Common trends

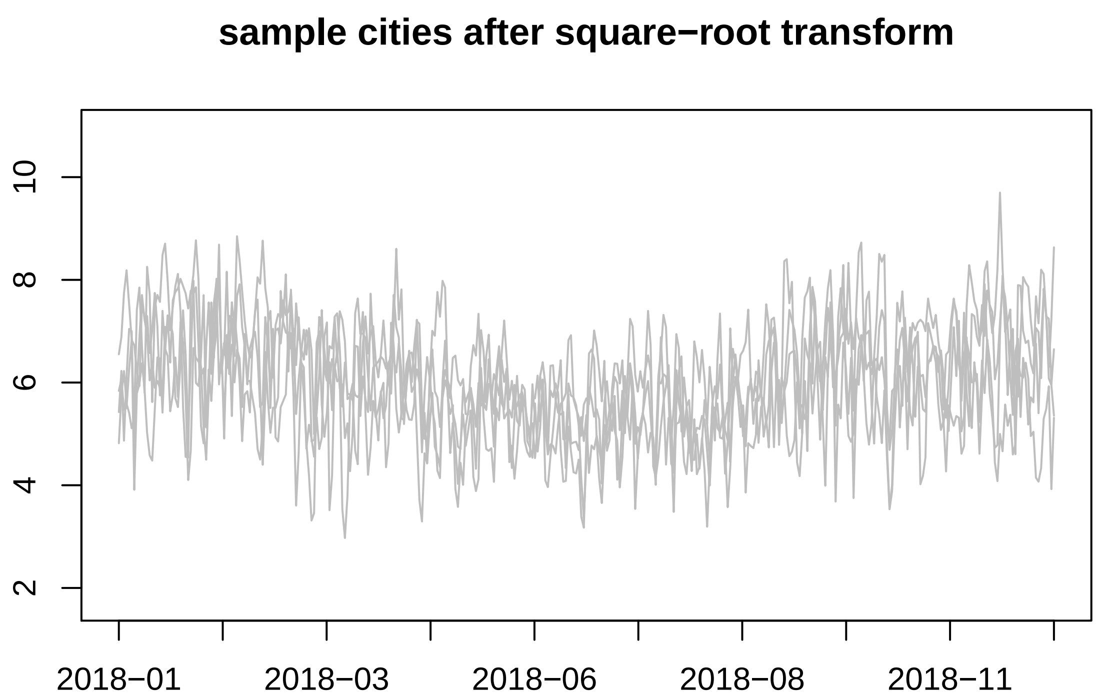
Seasonal trends in NO2 concentrations are well documented. With peaks generall occuring in Spring and Autumn.

```
par(mfrow = c(2,2))
sample.cities <- sqrt(sample.cities)
for (city in names(sample.cities)) plot.with.dates.axis(sample.cities[[city]], rownames(sample.cities),
```



Taking the square-root transform to bnring the data back to the light tailed domain as suggested in (**???**) and plotting the time series together the seasonal component is somewhat clearer.

```
df.plot.with.dates.axis(sample.cities, col = "grey", main = "sample cities after square-root transform")
```



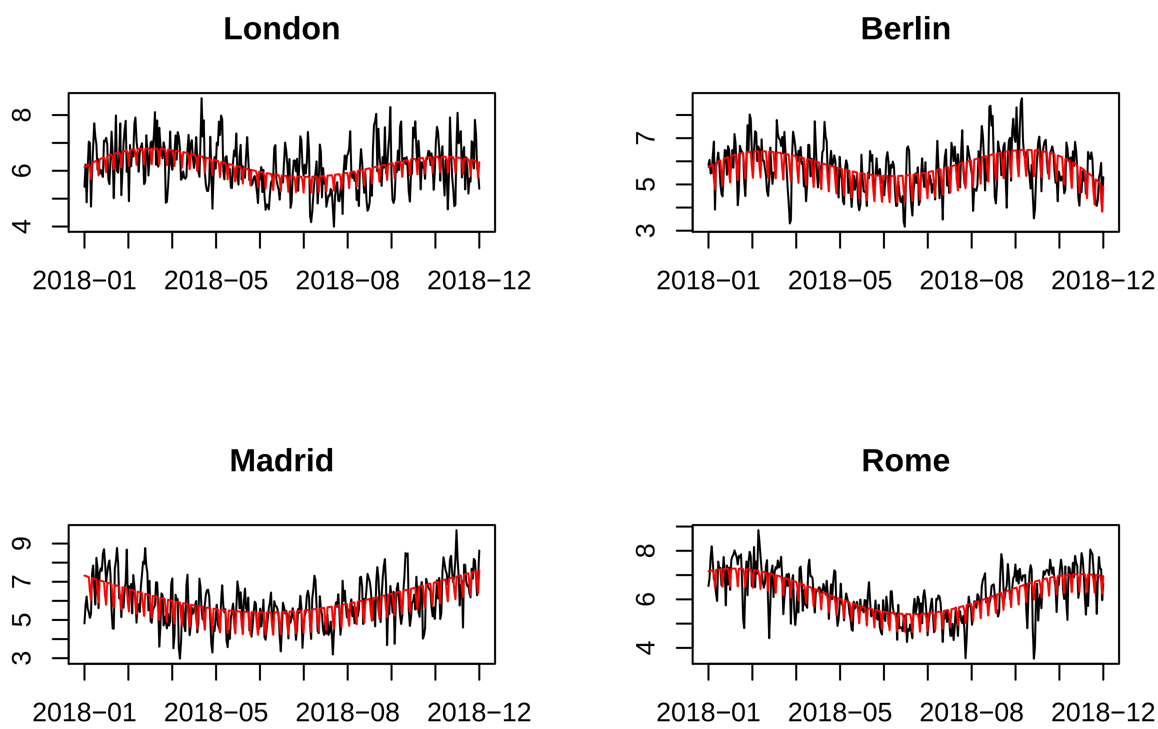
Removing trends

The trends discussed above can be eliminated by regressing onto Fourier frequencies corresponding to yearly and quarterly cycles, as well as indicators marking weekends.

```
days <- as.Date(rownames(sample.cities), format = "%Y-%m-%d")
tt <- 1:length(days)

sat <- ifelse(weekdays(days) == "Saturday", 1, 0)
sun <- ifelse(weekdays(days) == "Sunday", 1, 0)

s.qt <- sin(2*pi*tt/365/4)
c.qt <- cos(2*pi*tt/365/4)
s.yr <- sin(2*pi*tt/365)
c.yr <- cos(2*pi*tt/365)

trend.models <- list()
```

```r
for (city in names(sample.cities)) trend.models[[city]] <- lm(
    sample.cities[[city]] ~ sat + sun + s.qt + c.qt + s.yr + c.yr
    )
```

Visually the linear models seem to capture the seasonal component in the data well.



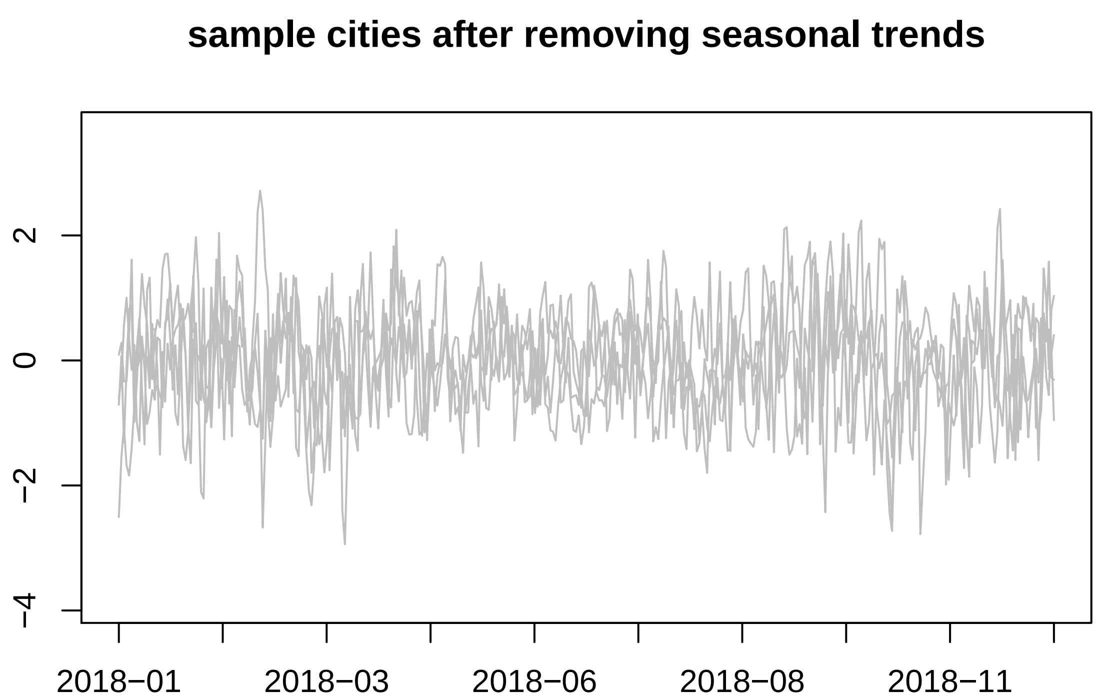
Further, for all cities in the sample the majority of regression coefficients are strongly statistically significant.

```r
summary(trend.models[["London"]])
```

```
##
## Call:
## lm(formula = sample.cities[[city]] ~ sat + sun + s.qt + c.qt +
##     s.yr + c.yr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.79655 -0.57263 -0.05911  0.55718  2.08863
##
## Coefficients:
```

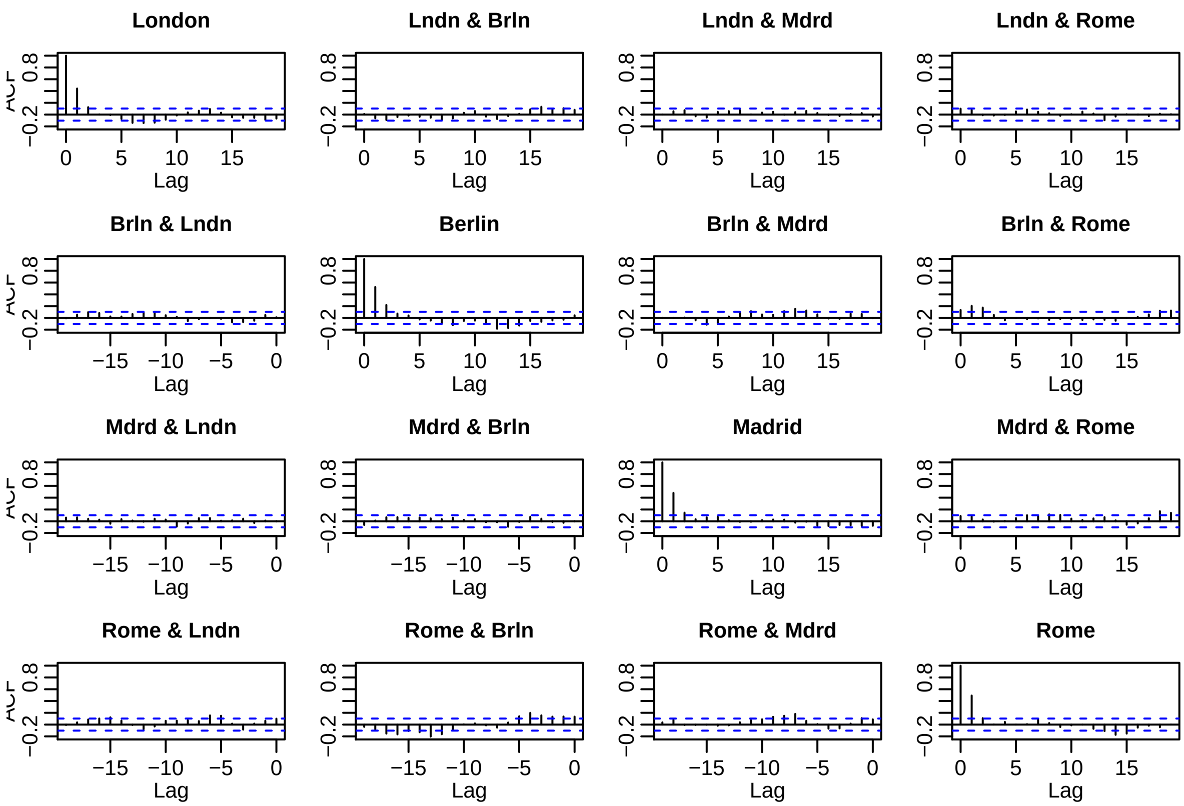
```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.12462    1.60486   0.701  0.48391
## sat         -0.11958    0.11961  -1.000  0.31811
## sun         -0.56415    0.11962  -4.716 3.45e-06 ***
## s.qt         4.15889    1.26774   3.281  0.00114 **
## c.qt         3.95777    1.26236   3.135  0.00186 **
## s.yr         0.30412    0.09646   3.153  0.00175 **
## c.yr         1.02802    0.22169   4.637 4.96e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7876 on 358 degrees of freedom
## Multiple R-squared:  0.1891, Adjusted R-squared:  0.1755
## F-statistic: 13.91 on 6 and 358 DF,  p-value: 3.106e-14
```

Finally, there is no visually obvious trend component in the empirical residuals from the above regression.



Removing serial correlation

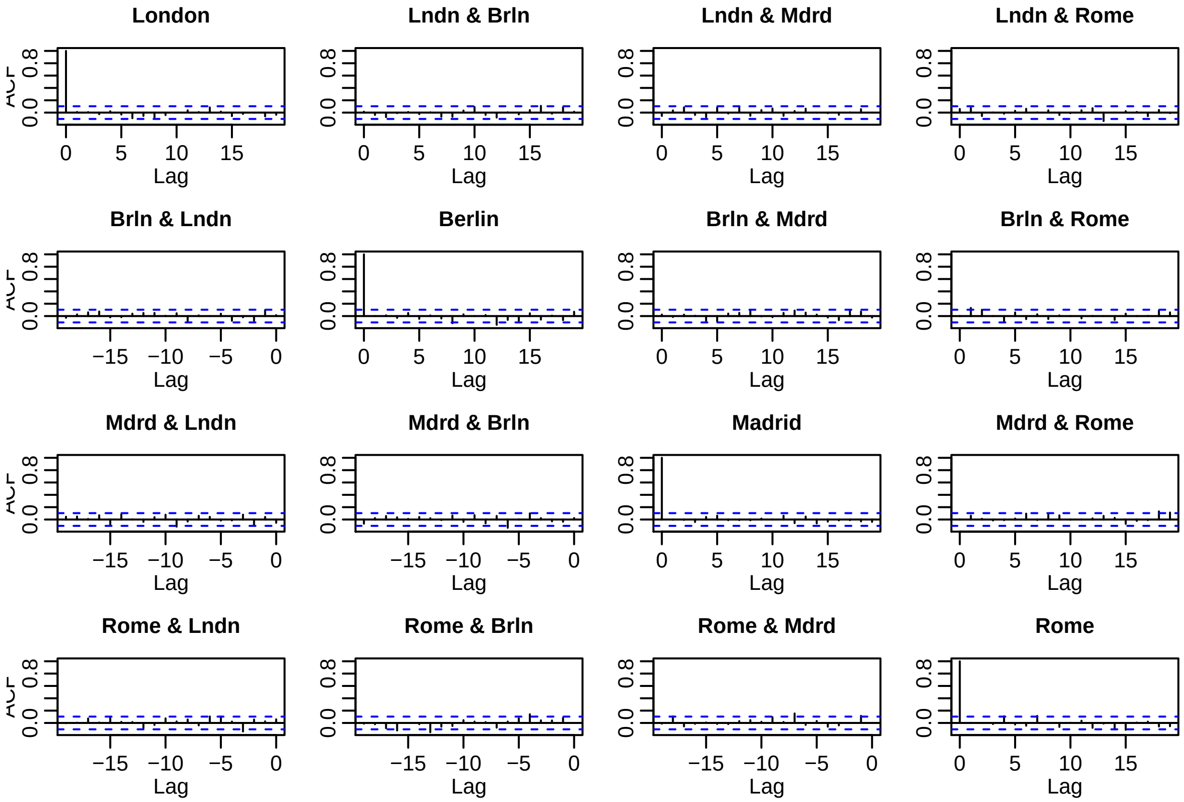
Although the seasonal component has been eliminated, the data thus processed is still auto correlated.



The data may be whitened by fitting an AR model and saving the residuals.

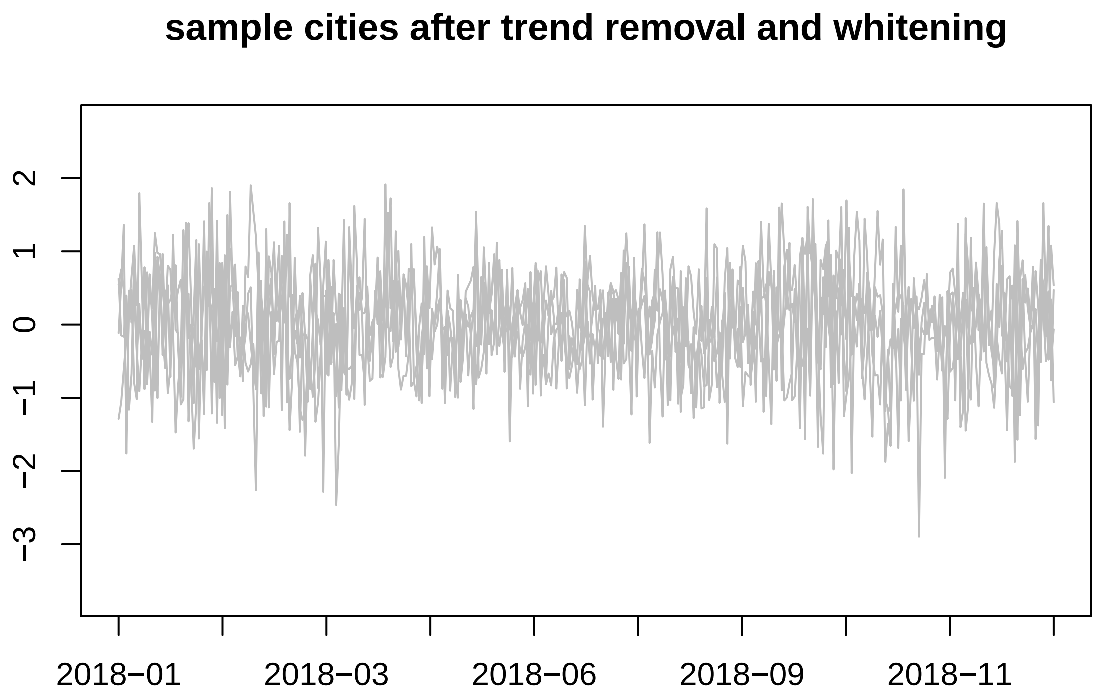
```
ar.models <- list()
for (city in names(sample.cities)) ar.models[[city]] <- ar(detrended.data[[city]], order.max = 3)
```

Whitening the data in this way seems to remove all series correlation.



Finally, the det-trended and whitened data is plotted below.

```
df.plot.with.dates.axis(whitened.data, col = "grey", main = "sample cities after trend removal and whit
```


sample cities after trend removal and whitening
2
1
0
−1
−2
−3
2018−01
2018−03
2018−06
2018−09
2018−11