Humboldt University
School of Business and Economics

Institute for Statistics

Report for Statistical Programming Languages

Theme:	Air Pollution in China	
Author:	Christoph Altmeppen	MatNr. 566640
Author:	Bjoern Bokelmann	MatNr. 551111
Author:	Thomas Dengler	MatNr. 565323
Author:	Benjamin Schwab	MatNr. 569347
Version of:	August 14, 2016	
1. Supervisor:	Elisabeth Bommes	
2. Supervisor:	Franziska Schulz	
3. Supervisor:	Christoph Schult	

Contents

1 Introduction

Since the end of the 1970s the Chinese population has experienced an unprecedented increase in living standards and economic development. This good news is dampened by the environmental consequences. Increasing levels of air pollution, especially particulate matter, and its effect on public health have been one of the most urgent political issues in China in the last years. Increased levels of particulate matter in the atmosphere have been found to increase rates lung cancer, respiratory and cardiovascular diseases [CRAO+05].

We try to contribute to existing research on this topic by combining several datasets and using the tools of data analysis on them to explore the structure of air pollution of Chinese cities and evaluate some hypothesis for their causes. In the first part, we test the long-run consequences of the Chinese heating policy which was established about half a century ago. In the second part, we try to find clusters in the data and perform a one-way ANOVA based on the clusters in the third part. In the last part, we visualize the data with help of the R-package "ggmap".

This intruduction and the first part has been written by Christoph Altmeppen, the second part has been written by Bjoern Bokelmann, the third part has been written by Thomas Dengler and the fourth part has been written by Benjamin Schwab. The code was tested and works with the latest version of R (3.3.1).

2 The Qin-Huai Line

2.1 Introduction

During the system of the planned economy in the three decades since 1950, a heating policy was established in China, which led to the provision of free heating to the households by the government during the winter months. However, this policy applies only to households in northern China. Since heating is mostly done by burning coal, it is a major contributor to pollution. The heating policy is not as relevant today as it was half a century ago since more and more households can afford heating by themselves, but it is a) still in place and b) has led to an increased installment of coal burning facilities in the north, which should lead to a higher degree of air pollution in northern China. This can be tested: An unofficial dividing line between northern and southern China is the line formed by the Huai River and the Qin Mountains. If the policy has a measurable effect, the air pollution should be higher in cities north of this line.

The Qin-Huai line can reasonably approximated by the 33rd parallel north. We can therefore use the geographic data of the maps package for R to test this hypothesis. We use the name variable to extract all relevant cities, keep the variables for latitude and longitude and merge these with our dataset.

2.2 Theory and Design

To test our hypothesis, we first use a Welsh test to test for a difference in the mean of PM10 pollution between the southern and the northern Chinese cities while allowing for different variances in the two samples. We proceed with a regression of the PM10 levels to control for possible economic confounding variables, like log(GDP), the share of secondary industry, log(Imports) and population size. Out model is a simple linear regression model:

$$\begin{aligned} PM10 = &log(GDP) + SecondaryIndustry + (Latitude > 33) \\ &+ log(Imports) + Population \end{aligned} \tag{2.2.0.1}$$

2.3 Implementation

First, we plot the latitude and the PM10 pollution for the cities and mark the Qin-Huai line by drawing a line which intersects the x-axis at 33 (line 53, appendix A), see figure 1. The respective means are calculated in lines 57 and 58 and drawn into the diagram in lines 61 and 62. We proceed with a t-test in line 66. The regression is done in line 71. We use the regression object as input for the stargazer command in line 75, which gives a formatted Latex-code of the regression output.

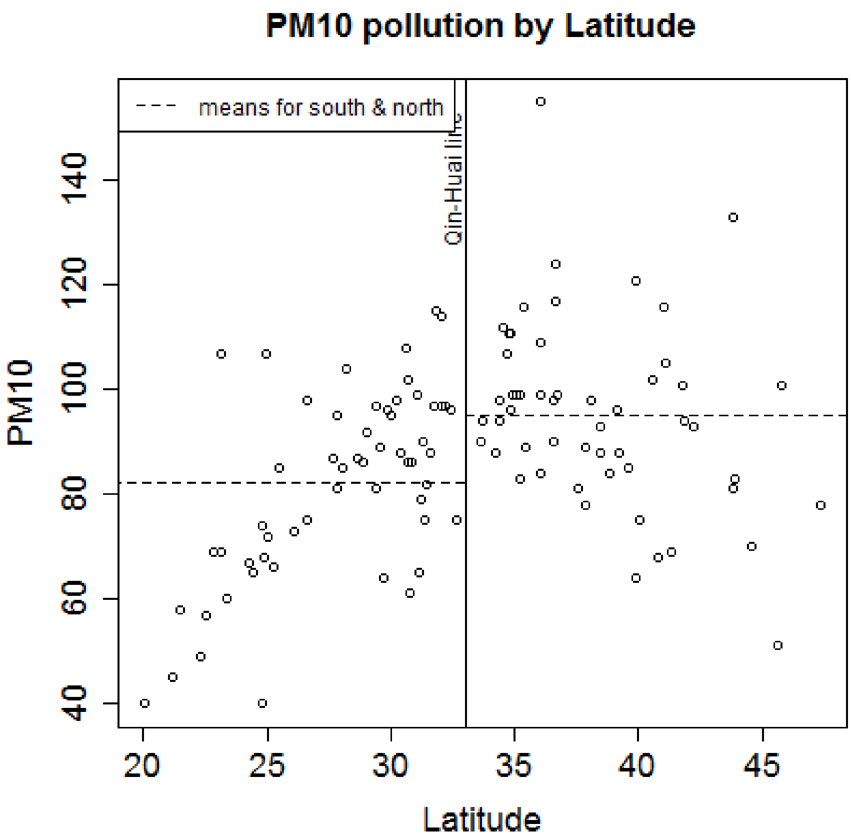

Figure 1: comparing the means of PM10 pollution in southern and northern China

2.4 Empirical Study

Our quantlet SPLac_China_QinHuai combines data about air pollution, geographical indicators and economic indicators of major Chinese cities, tests the historic influence of the so called Qin-Huai line and exports the output in Latex format.

We manually built a dataset with economic indicators using the data from China Knowledge Online [Chi14]. These data include:

Description	Variable
City name	Name
Land Area in Km^2	Area
Population in Million	Population
Share of primary industry (agriculture)	PrimaryIndustry
Share of secondary industry	SecondaryIndustry
Share of tertiary industry (services)	TertiaryIndustry
Gross Domestic Product in Bil. Yuan	GDP
Unemployment Rate	Unemployment
Fixed Asset Investments in Bil. Yuan	FixedAssetInvestment
Exports in Mil. USD	Exports
Imports in Mil. USD	Imports
Total Exports and Imports in Mil. USD	TotalExportsImports
Sales of Social Consumer Goods in Bil. Yuan	SalesSocialConsumerGoods

Table 1: Variable names and descriptions

The air pollution data are from the WHO Ambient outdoor air pollution database [Wor14]. It maintains the average annual air pollution in several years of major world cities measures in PM10 and PM2.5, which indicates the amount of particular matter in micrograms per cubic meter. The geographical data are from the R maps package. It maintains geographical data of major world cities, including latitude and longitude.

We first extracted the data about Chinese cities from the WHO air pollution dataset using the data.frame command, keeping only observations for which the third column in the dataframe indicated China. We then kept only the columns four and five with the city names and the PM10 pollution. For better tractability we assigned variable names (Name, PM10). After reading in the economic indicator data, we excluded observations 76, 89 and 94, which correspond to the cities Taian, Xi'an and Yan'an, because there were no data available for them. We are left with observations for 109 cities. The Name variable was then used for merging the data with the pollution data by the city names. The PM10 variable was converted to numeric. We decided not to use the PM2.5 data because for each observation only one of the PM variables was truly original in the WHO data, while the other variable was just converted by a fixed scalar. The PM10 data thus do not provide additional information. It should not matter for the analysis if one takes the one or the other variable. Both the China Knowledge Online data and the maps data provided information about population, but the data from CKO are

Dependent variable: PM10	
log(GDP)	15.84***
SecondaryIndustry	35.39**
Latitude >33	10.78***
log(Imports)	−4.34***
Population	−0.22
Constant	44.30***
Observations	109
Adjusted R^2	0.25

Table 2: Regression results. Note: *p<0.1; **p<0.05; ***p<0.01

from 2013 while the data from the maps package are from 2006. We therefore dropped the population variable left over from the maps package.

For later analysis we created a variable for population density by dividing the population by the area, indicating million inhabitants per square kilometer. We also calculated the absolute size of secondary industry by multiplying the share of secondary industry by the total GDP.

The results in table 2 support our hypothesis that northern Chinese cities have a higher amount of air pollution than southern Chinese cities. According to the Welsh t-test, the difference of the means is statistically significant at the 1% level (p-value = 0.0002191). The effect of the latitude is still significant at the 1%-level if we control for other factors in a simple linear regression model.

2.5 Conclusion

We combined data on PM2.5 air pollution with economic and geographical data for 109 Chinese cities to test a possible effect of the Chinese central heating policy, which applied only to northern Chinese Cities. We make use of the fact that the geographical division of northern and southern China can be reasonably approximated by the 33rd parallel north. In a linear regression which controls for economic factors a dummy for being north of the Qin-Huai line is positiv and significant. This indicates that the central heating policy established decades ago does have a significant influence on air pollution and consequentially on public health.

3 Cluster Analysis

3.1 Introduction

After analyzing common effects on air pollution for our 109 Chinese cities, we now try to obtain additional knowledge by categorizing these cities. We would expect great metropolitan regions to „behave“ different from rather remote cities in terms of air pollution and manufacturing areas „behave“ very different from highly developed cities with a big service sector.
In addition many geographical considerations are easier if each city is only represented by the number of the respective cluster instead of the 17 variables of our data set. For example we could generate maps and analyze differences in location of our clusters.

3.2 Theory and Design

Building clusters of objects from given data consists of two parts: The elaborate part is the preparation of the data which means selecting and modifying variables. This will be described in section 3.3.1.
The second part is the actual clustering algorithm. In our project we only apply hierarchical agglomerative clustering methods which is one of many classes of possible algorithms. These algorithms follow the procedure: First each element is in its own cluster (agglomerative). Then step by step, different clusters are merged together and once two elements are in the same cluster, they stay in the same cluster until the end (hierarchical). The next section briefly describes how these algorithms work.

3.2.1 Hierarchical Agglomerative Clustering

Each element a_i is represented by a tupel $(x_{a_1}, ..., x_{a_n})$ according to the chosen variables. First each element is considered as an independent cluster. Then we calculate a distance d_{ij} for each pair of clusters (a_i, a_j). Therefor we need to choose a metric according to the type of variables. For continuous variables the euclidean metric is most commonly used. This leads to the distance matrix $D = (d_{ij})_{ij}$.
In the next step, the clusters a_i, a_j with the lowest difference d_{ij} are merged into a joint cluster A. For A we then calculate the distance to any other cluster which leads to a modified distance matrix D'.
We repeat this step until we get the desired number of clusters.
The question arises, how to calculate the distance between two clusters that consist of multiple elements. Here several definitions of the distance are possible, leading to different fusion algorithms (see 3.4.1).

3.3 Implementation

3.3.1 Preparation of Data

Reduction of Variables The starting point of our analysis is a data set of 17 variables for each of the cities. So our interest lies in the reduction of variables. Because we want to get clusters that might explain the different outcomes in air pollution, the population density and the share of secondary industry are considered as the most relevant factors. Other variables are excluded due to high correlation with the former two variables or due to expected irrelevance.
An alternative would have been a reduction of variables by principle component analysis. But in order not to go beyond the scope of this project, we rather choose the former more simple solution.

Removal of Outliers The problem of outliers in the cluster analysis is that they differ so much from the other objects, that shorter differences become irrelevant in comparison. So we removed the (few) outliers in order to recognize structural differences between the (many) remaining cities. First we made boxplots:

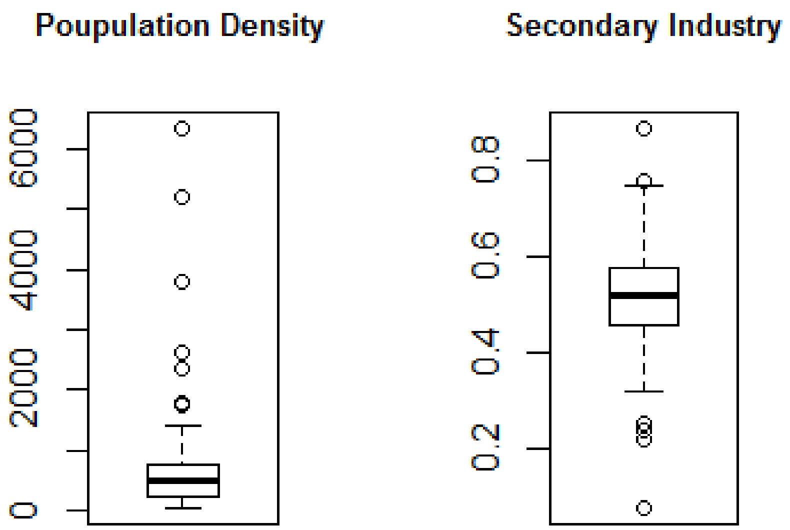

Figure 2: Boxplots of Population Density and Share of Secondary Industry

Obviously there are some outliers especially in population density. For the population density we choose $q_{0.75} + 1.5 \cdot IQR$ respectively $q_{0.25} - 1.5 \cdot IQR$ as upper respectively lower bound for inclusion. Because there are many cities with a slightly higher or lower share of secondary industry than the above given bounds, we shift the bounds in order to include as many cities in our analysis as possible (line 78).

```
1 # Get upper and lower bound
2 pop_ex_min       = boxplot.stats(PopulationDensity)$stats[1]
```

```
pop_ex_max       = boxplot.stats(PopulationDensity)$stats[5]
sec_ind_ex_min = boxplot.stats(SecondaryIndustry)$stats[1] * 4 / 5
sec_ind_ex_max = boxplot.stats(SecondaryIndustry)$stats[5] * 3 / 2
```

In section 3.4 we analyze the impact of including outliers into the cluster algorithm.

Centering of Variables Because the scales of our relevant variables differ significantly, we perform a z-transformation. If we would run the cluster algorithm on the unstandardized variables, the difference in share of secondary industry would be irrelevant in comparizon to the difference in population density. (The former difference is shorter than 1 while the latter is given in multiples of 1000.) We use the *scale-function* to standardize our variables (line 98).

```
# 3) Center Variables
cent_var = apply(X = as.matrix(data_cluster[, 2:7]), MARGIN = 2, FUN =
    scale)
```

3.3.2 The Cluster Algorithm

In order to perform the cluster algorithm, we use the library *cluster* (line 60). The actual cluster algorithm is performed in lines 110 to 124. The corresponding code is given below.

```
# Perform Hierachical Cluster Analysis using Euclidian Distance and
# Ward Fusion Algorithm

# Generate Distance Matrix using Euclidean Metric
dist_mat = dist(data_cluster[, c(2,3)])

# Perform clustering algorithm
cluster_tree = hclust(dist_mat)

# Dendrogram
plot(as.dendrogram(cluster_tree, hang = -1), main = 'Dendrogram Ward
    Algorithm', leaflab = "none")

# Choose certain number of clusters according to Dendrogram
num_clus = 4
Cluster   = cutree(cluster_tree, k = num_clus)
```

First the distance matrix is created using *dist* (line 114). The default metric of *dist* is euclidean, which is appropriate in our case, as we consider continuous variables.

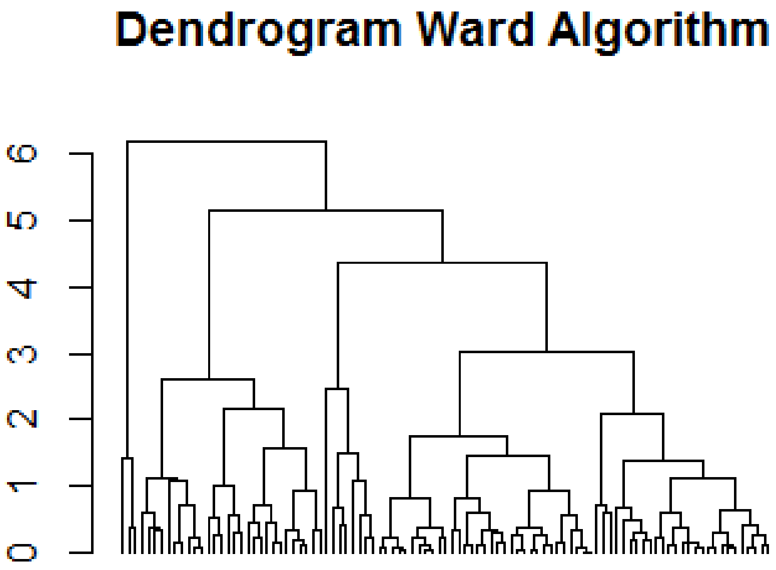

Figure 3: Dendrogram using Ward algorithm

Then *hclust* performs the hierarchical clustering algorithm, where different clusters are step by step merged together until all cities are in one cluster (see 3.2). *hclust* uses by default the Ward algorithm. In section 3.4.1 we will compare the result with other fusion algorithms.
Next we plot the dendrogram, which visualizes the merging process (see figure 3). The vertical axis represents the distance between two clusters. Now we need to choose the number of clusters. In order to get clusters that differ much, we stop the merging process at the longest vertical line, which leads to a number of four clusters.

Then the method *cutree* stops the merging process at the chosen number of clusters and provides the resulting clusters.

Resulting Clusters Figure 4 shows the result of the clustering process. Cluster 1 includes cities with a moderate population density and a moderate share of secondary industry. Cluster 2 can be interpreted as the industrial cities as they are characterized by a high share of secondary industry. Cluster 3 includes cities with a very high population density and a relatively low share of secondary industry. They might be highly developed cities with a lot of jobs in service sector. Finally Cluster 4 includes cities with a very low population density. They might be remote cities.

The formerly excluded cities with an extremely high population density are later included into a fifth (artificial) cluster (line 131). This is because some of these cities (like Hong Kong and Shanghai) are well known and thus we do not want to exclude them from our analysis.

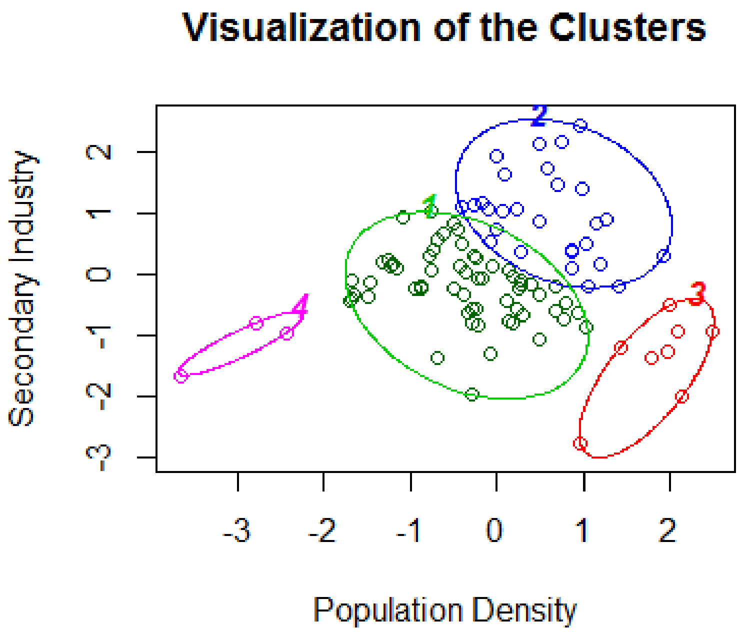

Figure 4: The Resulting Clusters

3.4 Empirical Study

The result of section 3.3 is generated by hierarchical clustering using the Ward fusion algorithm. A natural question that arises is whether this algorithm leads to the „optimal“ result. We evaluate the quality of a clustering in terms of the knowledge one can get about the structure of Chinese cities. A desirable result would be a low number of clusters, which differ significantly and which allow for plausible interpretation. We would also like to get clusters which do not differ to much in size.
To assess the quality of our result, we cluster our data with different algorithms and compare the results. But because there is a huge variety of clustering methods, we stay in the context of hierarchical agglomerative clustering and only compare different fusion algorithms (see section 3.4.1).
In section 3.3.1 we argued that outliers would compromise the clustering result. This will be tested in section 3.4.2.

3.4.1 Comparison of Fusion Algorithms

We consider the **single linkage** and the **centroid** fusion algorithms as alternatives to the **Ward algorithm**. They all merge clusters according to their respective distance to each other. The difference between these algorithms is due to the different definition of the distance between two clusters A and B. The **single linkage** takes the minimum of the distances between all possible pairs of elements from A and B. The **centroid**

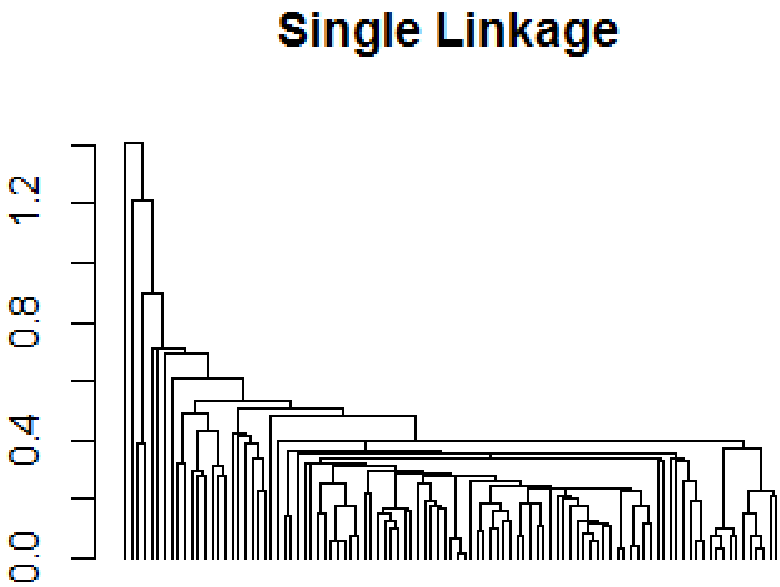

Figure 5: Dendrogram for Single Linkage Algorithm

algorithm calculates the difference between the average of A and B. Finally **Ward** expands the centroid distance by also taking into account the differences within the merged cluster of A and B.
Lines 143 to 166 correspond to the comparison. The resulting dendrograms are given in figures 5 and 6.
Following the criteria of maximum distance between clusters (see section 3.3.2) we would choose a number of three clusters for both algorithms. In both cases this would lead to one very big and two very small clusters which, according to the considerations at beginning of section 3.4, is of less value than the result obtained by the **Ward algorithm**.

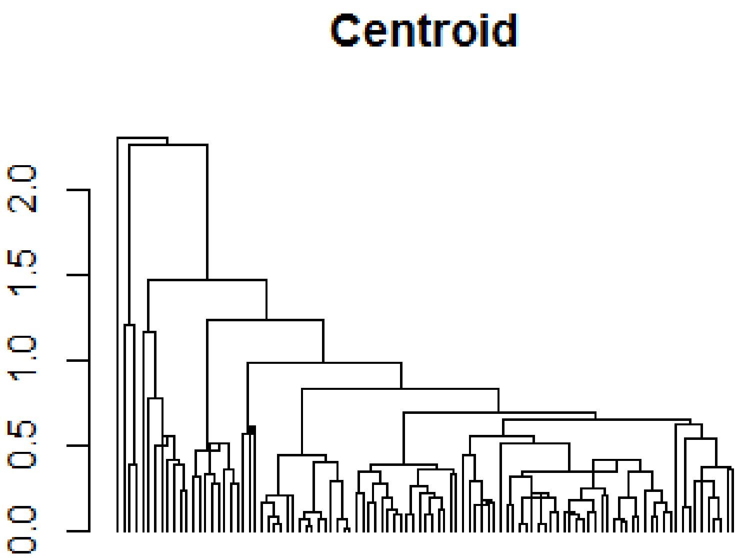

Figure 6: Dendrogram for Centroid Algorithm

3.4.2 The Impact of Outliers

In lines 167 to 202 clustering according to section 3.3.1 and 3.3.2 is performed but without the exclusion of outliers. Figure 7 shows the resulting clusters. As expected the big difference in population density of the outlier cities dominates differences between the rest of the cities and leads to rather useless clusters i.e. all cities except for three outliers are included into one big cluster.

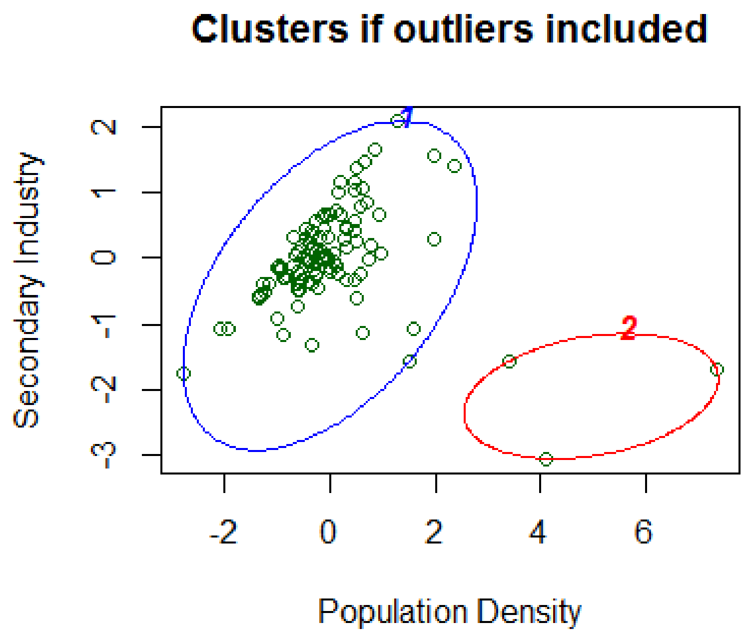

Figure 7: Resulting Clusters if outliers are included

3.5 Conclusion

3.5.1 Results

Given the data of 109 Chinese cities we performed cluster analysis using hierarchical agglomerative clustering methods. We choose population density and share of secondary industry as relevant variables for the clustering process and excluded outlier cities. We compared the three fusion algorithms **Ward**, **single linkage** and **centroid**. We choose the result of the **Ward algorithm** because it generated the most homogenous sized clusters. Overall we received four clusters that were interpreted as “industrial cities”, “highly developed cities”, “remote cities” and “cities of moderate population density and share of secondary industry”.

We further analyzed the impact of outliers on our clustering algorithm and found out that the big distance that outliers have to other elements, makes the algorithm neglect differences between the remaining elements and thus leads to a less valuable result.

3.5.2 Limitations

The result could possibly still be improved by considering more variables. A principle component analysis on all the given economic factors might generate valuable variables. In addition more data about the infrastructure (like number of cars per person or light rail vehicles) and technical level would have been a useful supplement. However all this ideas were taken into consideration but not implemented in order not to go beyond the scope of this project.

4 Extending the regression and testing for cluster differences

4.1 Introduction

The first part of this paper analyzed the effect of various regressors on air pollution in Chinese cities. In the second part, cluster analysis was applied to categorize these cities along two dimensions. The following part adds to our analysis in two ways. First, we include another variable, the distance to coal power stations, in our regression. Second, we test whether the level of air pollution is significantly different between clusters.

4.2 Theory and Design

After adding the new variable to the model, the simple linear regression looks as follows:

$$\begin{aligned} PM10 = & log(GDP) + SecondaryIndustry + (Latitude > 33) \\ & + log(Imports) + PopulationDensity + Distance \end{aligned} \tag{4.2.0.1}$$

where *Distance* represents the distance of a particular city to the nearest coal power station. In obtaining this variable, the key steps were transforming the raw data and then using a loop that assigns the closest coal station to each city.

In order to check whether the clusters show differences in air pollution, we start with simple a graphical comparison. Then we perform an analysis of variance (ANOVA) test in its simplest form to test whether the means of all the clusters are equal or not. It belongs to the category of Omnibus tests and as such the null hypothesis states that all the subgroup means are equal, whereas rejecting the null hypothesis implies that there is a difference between at least one of the groups,

$$H_0 : \mu_1 = ... = \mu_k \tag{4.2.0.2}$$

$$H_1 : \text{There exists at least one pair with } \mu_i \neq \mu_j \tag{4.2.0.3}$$

The reliability of ANOVA depends on two assumptions. First, the observations in each subgroup must be normally distributed. Second, the variances must be equal or at least of similar size. Appropriate tests for these assumptions will be presented, as well as a non parametric test as an alternative to ANOVA. Lastly, a pairwise t-test is applied to find differences between specific clusters.

4.3 Implementation

4.3.1 Creating the distance variable

The raw data taken from wikipedia[wik16] contains the name of the coal station and the coordinates. The coordinates are in degrees minutes second format. R requires decimal degrees, thus a transformation is necessary.

```
#Splitting the string into two at the blank space
long.lat        = as.data.frame(do.call(rbind, strsplit(stations$
    Coordinates, ' ')))
names(long.lat) = c("lat","long")
```

The example code above splits the column containing a string with both longitude and latitude into two separate columns. One with longitude and one with latitude. These are saved as a data frame in *long.lat*. The first argument for *strsplit* is a character vector and the second argument contains the element at which the split is meant to be performed, a blank space in this case. *rbind* combines the list of elements created by *strsplit* by rows. The function *do.call* is wrapped around it, requiring a function to call as its first argument (*rbind*) and a list of elements to assign this function to as its second argument. In this way, *rbind* combines the first half of the character string (latitude) and the second half (longitude) by rows in separate columns, resulting in the two desired variables.

Following that, the next code example converts the coordinates into decimal degrees. This is done by splitting each coordinate variable into separate columns. One for degrees ($dms[,1]$), one for minutes ($dms[,2]$) and one for seconds ($dms[,3]$). Minutes are divided by 60 and seconds are divided by 3600 and then added up, resulting in coordinates in the form of decimal degrees.

```
#Converting degrees, minutes, seconds into decimal degrees
dms      = do.call(rbind, strsplit(as.character(long.lat$lat), ":"))
long.lat$lat = as.numeric(dms[,1]) + (as.numeric(dms[,2]) +
as.numeric(dms[,3])/60)/60
rm(dms)

dms          = do.call(rbind, strsplit(as.character(long.lat$long), ":")
    )
long.lat$long = as.numeric(dms[,1]) + (as.numeric(dms[,2]) +
as.numeric(dms[,3])/60)/60
rm(dms)
```

The function *spDists* is applied to calculate the spherical distance between each city and each power station, resulting in a 102(cities)x50(stations) matrix. Then a loop is used that searches for the minimal distance(*which.min*) in each row and saves the corresponding power station in a variable (*closest*). The distance between each city and the closest station is saved in the variable *Distance*.

```
#convert into matrix format

cit  = as.matrix(cbind(data_china$Latitude,data_china$Longitude))
stat = as.matrix(long.lat)

#calculate spherical distances

dist = spDists(cit, stat, longlat=TRUE)
dist = round(dist, digits = 3)

#save closest station for each city and its corresponding distance in a
    variable

for (i in 1:102) {
  data_china[i,"closest"]  = stations$Station[which.min(dist[i,])]
  data_china[i,"Distance"] = min(dist[i,])
}
```

4.3.2 Cluster testing

An error bar diagram (figure 8) offers some perspective on whether there are significant differences in air pollution between the clusters. The circle indicates the estimated mean for each subgroup and the bars around it indicate the 95 percent confidence interval. Despite the fact that both the low number of observations and high amount of variance within lead to wide confidence intervals for cluster 4 and 5, differences between the clusters become apparent. In particular, cluster 3 and cluster 5 appear to have different means because the error bars do not overlap.

Before applying ANOVA, we tested for normality and for homogeneity of variances using the Shapiro-Wilk test and the Levene Test. The *tapply* function tells R to apply *shapiro.test* to each cell given by the value of the factor *Cluster*.

```
#Testing for normality and homogenity of variances
tapply(data_china$PM10, data_china$Cluster, shapiro.test)
leveneTest(data_china$PM10, data_china$Cluster)
```

Table 3 summarizes the Shapiro-Wilk test results. Normality is rejected at the 5 % level for cluster 2. Conversely, the Levene test cannot reject the null hypothesis of equal variances. With the assumption of normality violated, the use of a nonparametric

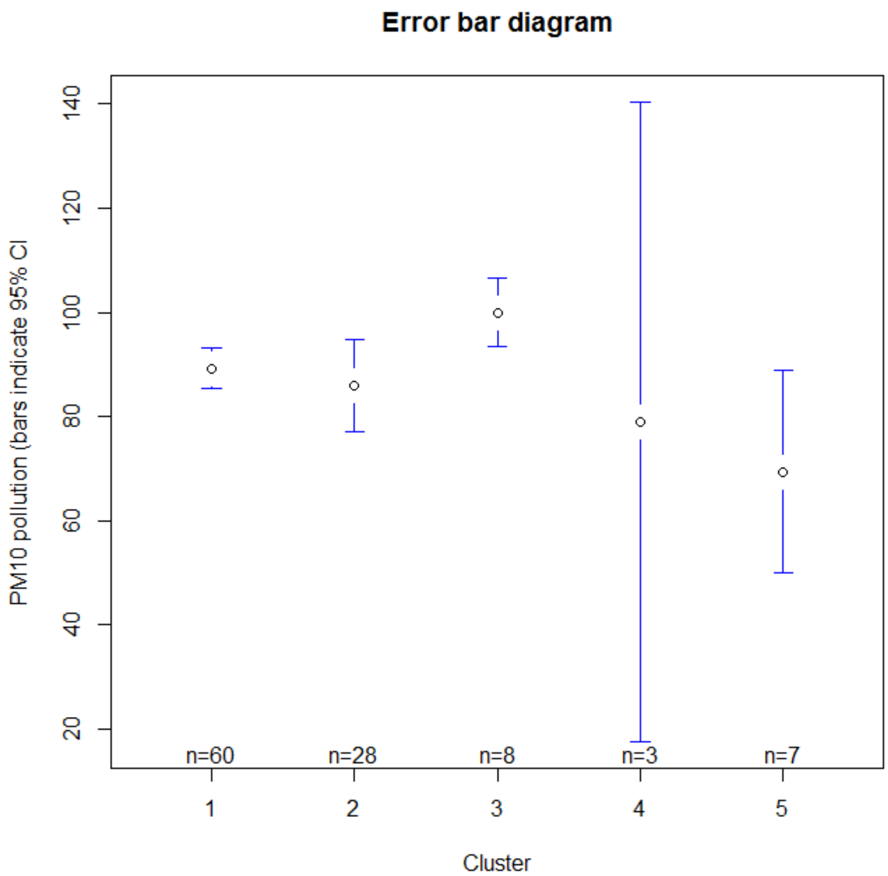

Figure 8: Comparing the PM10 air pollution between clusters

alternative might be more appropriate. The median test[1] is such an alternative, being a lot less reliant on assumptions while suffering from low test power. The code creates a table with two rows and as much columns as there are values for the factor variable z, in this case the variable *Cluster*. For each cluster, the observations that lie below and above the sample median are stored in row 1 and 2. The table is shown in the code example below. Then, a chi-squared independence test is applied to the table. The null hypothesis is that the clusters are independent of air pollution, meaning cell 1 and cell 2 must be of similar size in each column. Conversely, rejecting this hypothesis would suggest that the medians of the groups are not equal.

```
#create a mediantest
mediantest   = function(x, z){
  median     = median(x, na.rm=T)
  above      = (x > median)
mediantable = table(above, z)
  print(mediantable)
  chisq.test(mediantable)
}
mediantest(PM10, Cluster)
```

```
#look at the table
print(mediantable)
         z
above     1  2  3  4  5
  FALSE  28 17  1  2  6
  TRUE   32 11  7  1  1
```

Shapiro-Wilk Test	N	statistic	p-value
Cluster 1	60	0.974	0.242
Cluster 2	28	0.922	0.039**
Cluster 3	8	0.939	0.597
Cluster 4	3	0.901	0.388
Cluster 5	7	0.898	0.321
Levene Test	106	1.695	0.156

Table 3: Shapiro-Wilk Test and Levene Test

[1] code taken from Sigbert Klinke's lecture Datenanalyse 1

Dependent variable: PM10	
log(GDP)	13.72^{***}
SecondaryIndustry	42.16^{**}
PopulationDensity	-0.004
log(Imports)	-2.95^{*}
Latitude >33	13.60^{***}
Distance	-0.02^{***}
Constant	39.92^{***}
Observations	101
Adjusted R^2	0.28

Table 4: Regression results. Note: $^{*}p<0.1$; $^{**}p<0.05$; $^{***}p<0.01$

Omnibus tests	statistic	p-value
ANOVA	3.131	0.018^{**}
mediantest	9.923	0.042^{**}

Table 5: Omnibus test results

4.4 Empirical study/Testing

4.4.1 Regression

Coal plants are well documented producers of toxic emissions. One would expect that the further the distance to the next power plant, the lower the level of PM10 pollution in a city. The results in table 4 lend support to that hypothesis. the variable *Distance* is statistically significant at the 5 percent level (p-value = 0.001214).

4.4.2 Cluster testing

The results for the ANOVA and the median test are given in table 5. The corresponding code can be found in line 150-164. Note that for many of the tests it is important that the variable Cluster is stored as a factor. Otherwise, the Levene test would automatically coerce the variable to a factor , whereas ANOVA would produce incorrect output. The null hypothesis of equal means is rejected at the 5% significance level (p-value = 0.0179). Similarly, the median test suggests that the medians in the groups are different (p-value = 0.04175).

Now that we have confirmed that there are general differences in air pollution between the groups, we can apply a post-hoc test to find out which specific clusters are different. One such test that R offers is the pairwise t-test. Knowing that some of our subgroups are rather small, we chose the option $pool.sd = TRUE$. The function now calculates a common standard deviation for all groups and uses that for all the comparisons. In addition , we chose a bonferroni correction to counteract the errors that occur with multiple comparisons.

Cluster	1	2	3	4
2	1.000			
3	1.000	0.517		
4	1.000	1.000	0.857	
5	0.066*	0.316	0.013**	1.000

Table 6: Pairwise t-test

```r
#pairwise t test
pairwise.t.test(PM10, Cluster, p.adjust.method = "bonferroni",pool.sd =
    TRUE))
```

In this case 10 comparisons are made, therefore the p-values are adjusted with $p_i^* = p_i * 10$. The test results in table 6 show that only cluster 3 and 5 are different at the 5% significance level (adjusted p-value = 0.013). This is in line with the graphical analysis. To check for robustness, we performed the Tukey-HSD test and the Scheffé test, both yielding the same result. Remember that cluster 5 is the artificial cluster created for population density outliers, whereas cluster 3 is characterized by high population density and a low share of secondary industry, which we interpret as highly developed cities with a large service sector. The fact that the "highly developed" cluster has the highest mean pollution might appear counterintuitive at first. However, one could assume that governments of rich cities in China, contrary to more developed countries, have not yet made enough efforts to counteract the positive relationship between economic growth and air pollution. In conclusion, clustering along the dimensions of population density and share of secondary industry does not yield drastic differences in the level of air pollution. This is easily explained. First, the regression results show that only the secondary industry variable is significantly correlated with air pollution. Second, only two dimensional clustering was applied. It is reasonable to assume that clustering with more dimensions or factors created via PCA, all of them being correlated with PM10, would yield a different result.

4.4.3 Conclusion

In part 2, we arrived at the conclusion that the dataset could be extended by adding many more interesting variables. In this chapter we built on that idea and added a new variable capturing the distance to potential sources of particle matter pollution. Key steps were explained in detail: the recoding of coordinates into a decimal degrees format and the loop that assigns the closest power station to each city. The variable was found to have a significant impact on air pollution in a linear regression. In addition to that, this part applied the statistical tools that R offers for subgroup analysis to the clusters obtained in the previous chapter. In a way, this exercise lends additional support to the

OLS results. When one chooses variables for clustering that are only partly correlated with air pollution, one can expect merely coincidental differences with respect to that variable in the resulting subgroups. Thus, the results confirmed expectations.

5 Visualisation of spatial data using "ggmap"

5.1 Introduction

A visualisation of spatial data with the help of maps can provide additional information about the underlying data. What if the clusters were concentrated in some regions, could there exist an additional, geographical component that has been neglected so far? In addition, maps often help to create a better access to the data and make it therefore more tangible for the audience.

There are numerous different ways of creating map-plots with *R* that all have their genuine advantages and weaknesses. In this section the decision has been made in favor of the package *ggmap* written by David Kahle and Hadley Wickham. It provides an access to static maps from various online sources such as *Google Maps*, *OpenStreetMap* and *Stamen Maps* and, in addition, is relatively easy to use.[KW13]

5.2 Preparations

The first step is reading the dataset that has been saved as a csv.-file via the `read.csv()`-function and store it. We remove all other columns then the ones needed for our purposes and save the dataframe as `data_final`. In the next step the different clusters will be assigned with (clearly legible) colors by using a nested `ifelse()`-function. For the sake of convenience we use `attach()` wich will make the code significantly shorter and easier to read.

```
> data_china = read.csv("data_china.csv")
>
> data_final = data_china[,c(2,3,18,19,20)]
>
> # mapping Cluster numbers to different colours
> data_final$Colour = ifelse(is.na(data_final$Cluster) == 1, "grey",
    ifelse(data_final$Cluster == 1, "darkgreen", ifelse(data_final$Cluster
     == 2, "blue", ifelse(data_final$Cluster == 3, "red", ifelse(data_
    final$Cluster == 4, "magenta","brown")))))
>
> attach(data_final)
```

A short look at the dataset using `head()` confirms that the transformation was successful.

```
> head(data_final)
     Name PM10 Latitude Longitude Cluster    Colour
1  Anshan  105    41.12    122.95       1 darkgreen
2  Anyang  109    36.08    114.35       1 darkgreen
3 Baoding   84    38.87    115.48       1 darkgreen
```

```
4    Baoji    98    34.38    107.15    1  darkgreen
5    Baotou  102    40.60    110.05    1  darkgreen
6    Beihai   58    21.48    109.10    2       blue
```

We now install the required "ggmap"-package and load it.

```
> install.packages("ggmap")
>
> library(ggmap)
```

5.3 Quantlets

5.3.1 SPLsbChina_map1

The first quantlet SPLsbChina_map1 produces a Google-Map of China that depicts all those Chinese cities that are included in our dataset. The points indicating the location of the cities are in the respective color that was assigned to their cluster in Figure 4. The extreme values that were removed from the analysis are shown in brown color, those cities that were not included in any cluster are depicted in gray.

As a first step we first create an auxiliary map of class `ggmap` called `map_aux` via the `get_map()`-command. The predefined argument for location, `"China"` combined with a zoom-level of 4 is sufficient to depict all our cities, so there is no need for a further manual modification. Since we wanted to work with GoogleMaps the `source` parameter is `"google"`, the maptype `"roadmap"` is suitable for our purposes. The function `ggmap()` plots our auxiliary map. The `base_layer` argument must be filled with the output of the `ggplot()`-function that processed our dataset `data_final`. We store our raw map in the variable `map_raw`.

```
> map_aux = get_map(location = "China", maptype = "roadmap", zoom = 4,
    source = "google")
>
> map_raw = ggmap(map_aux, base_layer = ggplot(data_final))
```

The function `geom_point()` now adds points to our cities' geographical coordinates (Longitude and Latitude), for the argument `color` we use our predifined vector, `Colour`, that contains the colors of the respective clusters. With `labs()` we create a title and assign names the x- and y-axis. Please note that elements that were added to our raw map with a `+`-sign. We save the plot we created as `map_cities` and use the `print()`-command to show the map.

```
> map_cities = map_raw + geom_point(color = Colour, size = 1, aes(x =
    Longitude, y = Latitude))
>
> map_cities = map_cities + labs(title = "Cities by Cluster", x = "
    Longitude", y = "Latitude")
>
>print(map_cities)
```

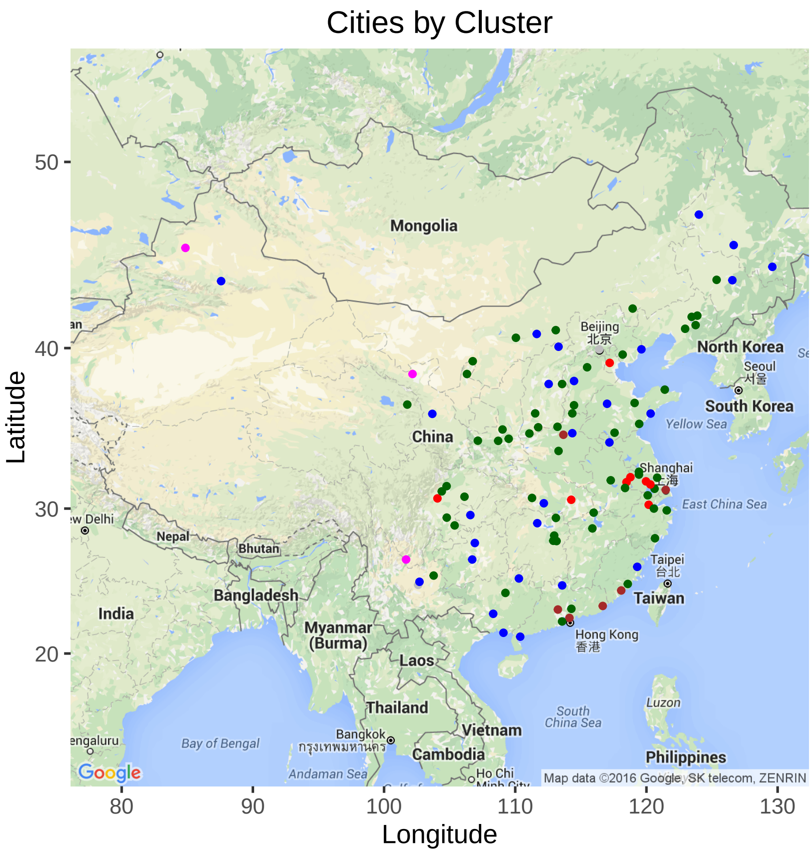

Figure 9: Quantlet SPLsbChina_map1 - Chinese Cities by Cluster

An interesting salience in this map is that many cities of the "red" cluster, with high population density and a low share of secondary industry, seem to be geographically concentrated around the area of Shanghai, whereas all three members of the "pink" cluster (both low population density and low share of secondary industry) are located in the Chinese hinterland, far from the coastal areas.

5.3.2 SPLsbChina_map2

Quantlet SPLsbChina_map2 produces another map of China which has two main features.

1. The size of the bubbles that are used to depict our cities is relative to the amount of PM10 a city emits

2. A blue semi-transparent rectangle marks the area around Shanghai

We again use our `map_raw` object as a base layer and plot the cities in a similar manner as in the preceding quantlet. There are, however, three alterations in the parameter-values that are necessary for the creation of our map. Firstly the color argument is simply changed to `"red"`, since we no longer want to distinguish between the different clusters. Most importantly `size` is not assigned a constant value anymore but the vector `PM10`. Therefore the size of the points is now relative to the corresponding value of PM10 emission of the city. The `alpha`-argument is added to manipulate the opacity of the points.

```
# Bubbles of the cities scaled by PM10
> map_pm10 = map_raw + geom_point(aes(x = Longitude, y = Latitude, size =
    PM10), color = "red", alpha = 0.4)
> map_pm10 = map_pm10 + labs(title = "Air Pollution in Chinese cities", x
    = "Longitude", y = "Latitude", size = "average PM10 ug/m3")
```

In the second step a semi-transparent rectangle of the area of our interest is added to the "bubble"-map we just created. Analogous to the `geom_point()`-function we hereby use `geom_rect()` which draws a rectangle with the corners (`xmin`, `xmax`, `ymin`, `ymax`). We include the geographic coordinates of the meridian and parallel arcs such that the area of greater Shanghai "fits" in our selection. Finally we choose a very low `alpha`-value for an increased transparency.

```
> map_rect = map_pm10 + geom_rect(aes(xmin = 116, xmax = 124, ymin = 28,
   ymax = 34), color = "slateblue3", alpha = 0.005)
> print(map_rect)
```

As we can see in Figure 10, without any further ado `ggmap()` automatically adds a legend that contains different threshold values of PM10 and the corresponding size of the bubbles starting from a value of 50 up to 150.

The picture also nicely depicts the geographical distribution of the biggest Chinese cities in general and those that have very high levels of PM10-emissions in particular. Especially in the Shanghai-area there seems to be a high density of high-level PM10-emitting cities, which serves as a reason for highlighting it in the map.

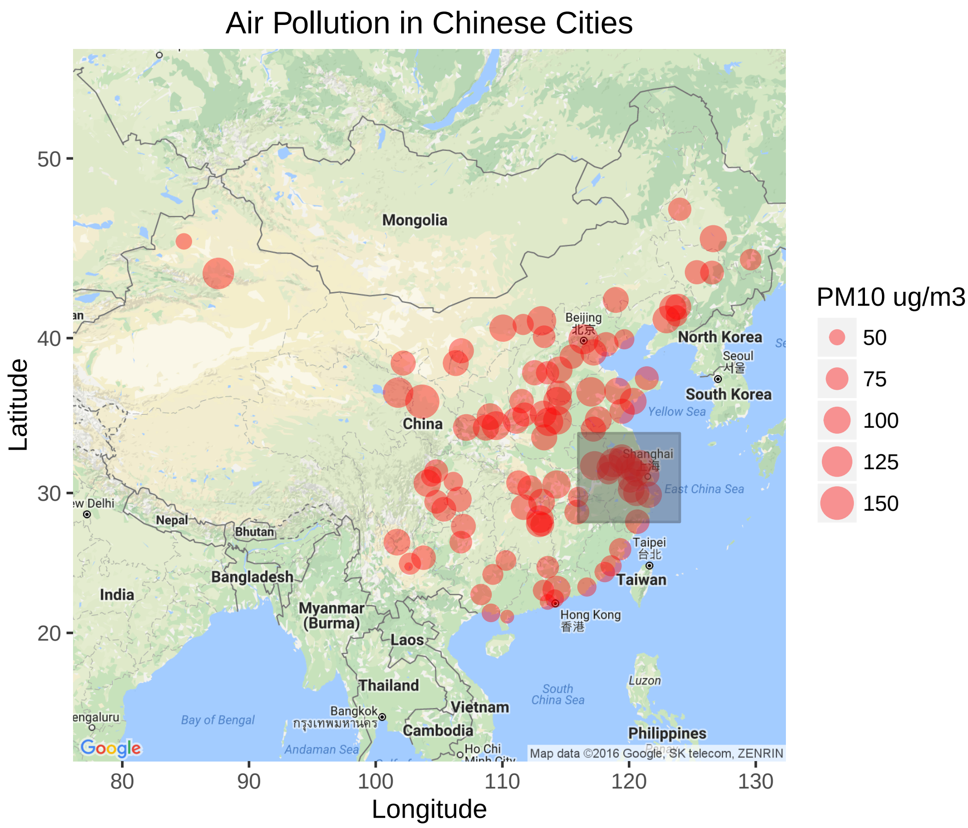

Figure 10: Quantlet 2 SPLsbChina_map2 - Bubble map scaled by PM10 emission

5.4 Conclusion

The *R*-package *ggmap* has proven to be a great way to conveniently create map plots. With only few lines of code complex graphics can be created that help to facilitate the analysis of spatial data. Potentially useful additional information can be gathered by regarding the geographic location of the cities. This could possibly help to make the composition of the different clusters more intuitive or provide a better understanding of the geographical distribution of PM10 emitting cities.

6 Conclusion

We combined environmental, economical and geographical data about Chinese cities to find patterns, analyze the relationship between those data and illustrate them graphically.

A simple linear regression indicates that the level of PM10 pollution in northern China is significantly higher than in southern China, which is probably a lasting impact of the Chinese heating policy.

We found out that a certain number of cities has an extreme high population density. Their impact as outliers on our clustering results was evaluated and as a consequence they were excluded from the clustering algorithm. We obtained four clusters according to population density and share of secondary industry. A fifth cluster includes the cities with an extreme high population density.

Furthermore, after adding another variable to our baseline regression, the distance to coal power plants, we obtained the expected result that the farther away a city lies from a coal power station, the lower the average level of PM10 air pollution.

In addition to that, we tested for differences in air pollution between our clusters, and found a significant difference only between cluster 3 and 5. We suspect that this result is due to insufficient correlation between population density and air pollution.

The *ggmap*-package has turned out to be a suitable choice for visualising our data and the map-plots that we created are a good supplement for the analysis of the underlying data. Further information such as geographical patterns of our predefined clusters or an apparent spatial concentration of high-level PM10-emitters in the Shanghai area can be obtained and leaves space for further investigations.

Literature

[Chi14] China Knowledge Online: Major Economic Indicators. (2014). http://www.chinaknowledge.com/CityInfo/CityInfo.aspx

[CRAO+05] Cohen, Aaron J. ; Ross Anderson, H ; Ostro, Bart ; Pandey, Kiran D. ; Krzyzanowski, Michal ; Künzli, Nino ; Gutschmidt, Kersten ; Pope, Arden ; Romieu, Isabelle ; Samet, Jonathan M. u. a.: The global burden of disease due to outdoor air pollution. In: *Journal of Toxicology and Environmental Health, Part A* 68 (2005), Nr. 13-14, S. 1301–1307

[Hla15] Hlavac, Marek ; Harvard University (Hrsg.): *stargazer: Well-Formatted Regression and Summary Statistics Tables*. Cambridge, USA: Harvard University, 2015. http://CRAN.R-project.org/package=stargazer. – R package version 5.2

[Kli15] Klinke, Sigbert: Datenanalyse I. (2015)

[KW13] Kahle, David ; Wickham, Hadley: ggmap: Spatial Visualization with ggplot2. In: *The R Journal* 5 (2013), Nr. 1, 144–161. http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf

[RABTPMD16] Richard A. Becker, Original S. b. ; Thomas P Minka, Allan R. Wilks. R. b. ; Deckmyn., Alex: *maps: Draw Geographical Maps*, 2016. https://CRAN.R-project.org/package=maps. – R package version 3.1.1

[wik16] wikipedia: List of power stations. (2016). https://en.wikipedia.org/wiki/List_of_power_stations_in_China

[Wor14] World Health Organization: Ambient Air Pollution Database. (2014), May. http://www.who.int/phe/health_topics/outdoorair/databases/cities/en/

7 Appendix A

```r
install.packages("readxl")
install.packages("maps")
install.packages("stargazer")

library(readxl)
library(maps)
library(stargazer)

# Loading and preparing data for our analysis
setwd("C:/Users/sgtpeppers/Dropbox/Statistical Programming Languages") # set working directory here
data              = read_excel("AAP_PM_database_May2014.xls", sheet = 2)
data_china        = data.frame(data[which(data[, 3] == "China"), ])
rm(data)
data_china        = data_china[, c(4, 5)]
names(data_china) = c("Name", "PM10")
# read in economic indicators
econ            = read_excel("economic_indicators.xls")
econ            = econ[-c(76, 89, 94), ]  # delete cities without entries
names(econ)[1]  = "Name"

# merge
data_china      = merge(data_china, econ, by = "Name")
rm(econ)

# convert PM variables to numeric
data_china$PM10 = as.numeric(data_china$PM10)

# Only consider values for population of econ data
data_china$Population = NULL

# Converteing values for GDP in Euro
data_china$gdp = data_china$gdp * 0.136

# calculate pop density
data_china$popDens = data_china$pop/data_china$area * 10^6

# calculate size of secondary industry
data_china$indus  = data_china$secondaryInd * data_china$gdp
names(data_china) = c("Name", "PM10", "Area", "Population", "GDP", "PrimaryIndustry", "SecondaryIndustry", "TertiaryIndustry", "Unemployment", "FixedAssetInvestment", "TotalExportsImports", "Imports", "Exports", "SalesSocialConsumerGoods", "PopulationDensity", "TotalSecondaryIndustry")
# China Map with our selected cities get lon and lat out of map data
our_cities_map    = world.cities[which(is.element(world.cities$name, data_china$Name) & world.cities$country.etc == "China"),]

# merge with our data
our_cities        = our_cities_map[, c(1,4,5)]
rm(our_cities_map)
names(our_cities) = c("Name", "Latitude", "Longitude")
data_china        = merge(data_china, our_cities, by = "Name")
rm(our_cities)

png(filename = "plot.png")
# Qin-Huai line
```

```
plot(data_china$Latitude, data_china$PM10, main = "PM10 pollution by
    Latitude", ylab = "PM10", xlab = "Latitude")
abline(v = 33)
text(32.5, 140, "Qin-Huai line", srt= 90)

# calculate means
meanSouth = mean(data_china$PM10[data_china$Latitude < 33])  #82.1
meanNorth = mean(data_china$PM10[data_china$Latitude > 33])  #95.2 -->
    they differ!

# draw means and legend
lines(x = c(0, 33), y  = c(meanSouth, meanSouth), lty = "dashed")
lines(x = c(33, 50), y = c(meanNorth, meanNorth), lty = "dashed")
legend("topleft", legend="means for south & north", col = "black", lty
    =2)

dev.off()

# ttest
t.test(data_china$PM10 ~ as.factor(data_china$Latitude > 33))

# run regressions
lm = lm(PM10 ~ log(GDP) + SecondaryIndustry + (Latitude > 33) + log(
    Imports) + Population, data = data_china)

summary(lm)

stargazer(lm, font.size = "small", no.space = T, intercept.bottom = T,
    single.row = T, keep.stat = c("n","adj.rsq"), digits = 2, align = T,
    report = "vc*")
```

8 Appendix B

```
# Load and prepare data for our analysis
library(readxl)
library(maps)
setwd("C:/Users/BjÃ¶rn/Documents/Uni/MasterSem2/StatisticalProgramming/
    AirPolution")
data = read_excel("AAP_PM_database_May2014.xls", sheet = 2)
data_china = data.frame(data[which(data[, 3]=='China'),])
rm(data)
data_china = data_china[, c(4,5)]
names(data_china) = c('Name', 'PM10')

# Read in economic indicators
econ              = read_excel("economic_indicators.xls")
econ              = econ[-c(76,89,94),] #Delete cities without entries
names(econ)[1] = "Name"

# Merge
data_china = merge(data_china,econ,by='Name')
rm(econ)

# Convert PM variables to numeric
data_china$PM10 = as.numeric(data_china$PM10)

# Only consider values for population of econ data
data_china$Population = NULL

# Converte values for GDP in Euro
data_china$gdp = data_china$gdp * 0.136

# Calculate pop density
data_china$popDens = data_china$pop/data_china$area * 10^{6}

# Calculate size of secondary industry
data_china$indus  = data_china$secondaryInd * data_china$gdp
names(data_china) = c('Name', 'PM10', 'Area', 'Population', 'GDP', '
    PrimaryIndustry', 'SecondaryIndustry'
                        , 'TertiaryIndustry', 'Unemployment', '
                            FixedAssetInvestment', 'TotalExportsImports',
                        'Imports', 'Exports', 'SalesSocialConsumerGoods', '
                            PopulationDensity', 'TotalSecondaryIndustry')

# China Map with our selected cities get lon and lat out of map data
our_cities_map = world.cities[which(is.element(world.cities$name, data_
    china$Name) & world.cities$country.etc == "China"),]

# Delete small cities with the same name as big cities
our_cities_map = our_cities_map[-which(duplicated(our_cities_map$name)),]

# Merge with our data
our_cities = our_cities_map[, c(1,4,5)]
rm(our_cities_map)
names(our_cities) = c("Name", "Latitude", "Longitude")



data_china = merge(data_china, our_cities, by = "Name", all=TRUE)
rm(our_cities)
```

```
#############################################################################
#############################################################################
#############################################################################
#Hierachical Cluster Analysis according to economic factors

library(cluster)

# Preperation for Cluster Analysis

# 1) Select Variables for our data frame
attach(data_china)
gdp_percap           = GDP / (Population * 10^{6}) * 10^{9}
data_cluster         = cbind.data.frame(Name, PopulationDensity, gdp_
    percap, SecondaryIndustry,
                                   Unemployment, FixedAssetInvestment,
                                       SalesSocialConsumerGoods)
names(data_cluster) = c('Name', 'PopulationDensity', 'GDPperCapita', '
    SecondaryIndustry',
                       'Unemployment', 'FixedAssetInvestment', '
                          SalesSocialConsumerGoods')
detach(data_china)

# 2) Search for outliers
attach(data_cluster)
boxplot(PopulationDensity, main='Poupulation Density', sub='Detaction of
    Outliers')
boxplot(SecondaryIndustry, main='Share of Secondary Industry', sub='
    Detaction of Outliers')

# Get upper and lower bound
pop_ex_min     = boxplot.stats(PopulationDensity)$stats[1]
pop_ex_max     = boxplot.stats(PopulationDensity)$stats[5]
sec_ind_ex_min = boxplot.stats(SecondaryIndustry)$stats[1] * 4 / 5
sec_ind_ex_max = boxplot.stats(SecondaryIndustry)$stats[5] * 3 / 2

outlier_cities = which(PopulationDensity<pop_ex_min |PopulationDensity>
    pop_ex_max
                      |SecondaryIndustry<sec_ind_ex_min|SecondaryIndustry>
                          sec_ind_ex_max)

# Save cities with extrem high population
cluster_extpop = which(PopulationDensity > pop_ex_max)

# Save data including outliers for later analysis
data_cluster_outl = data_cluster

# Delete outliers
data_cluster=data_cluster[-outlier_cities,]
detach(data_cluster)

# 3) Center Variables
cent_var = apply(X = as.matrix(data_cluster[, 2:7]), MARGIN = 2, FUN =
    scale)


# 4) Generate Correlation Matrix
cor_mat = cor(cent_var, use = 'pairwise.complete.obs')
```

```
# 5) Generate modified data frame for Cluster Analysis
data_cluster         = cbind.data.frame(data_cluster$Name, cent_var)
data_cluster         = data_cluster[, c("data_cluster$Name", "
    PopulationDensity", "SecondaryIndustry")]
names(data_cluster) = c('Name', "PopulationDensity", "SecondaryIndustry")

########################################################################
# Perform Hierachical Cluster Analysis using Euclidian Distance and
# Ward Fusion Algorithm

# Generate Distance Matrix using Euclidean Metric
dist_mat = dist(data_cluster[, c(2,3)])

# Perform clustering algorithm
cluster_tree = hclust(dist_mat)

# Dendrogram
plot(as.dendrogram(cluster_tree, hang = -1), main = 'Dendrogram Ward
    Algorithm', leaflab = "none")

# Choose certain number of clusters according to Dendrogram
num_clus = 4
Cluster  = cutree(cluster_tree, k = num_clus)

# Add Cluster Information to China Data
data_cluster = cbind(data_cluster, Cluster)
name_cluster = data_cluster[, c('Name', 'Cluster')]
data_china   = merge(data_china, name_cluster, by = 'Name', all = TRUE)

# Cities with very high popluation density are included in Cluster 5
data_china$Cluster[cluster_extpop] = 5

# Visualizing Clustering of Data
colors         = c('darkgreen','blue','red','magenta')
colors_cluster = colors[data_cluster$Cluster]
clusplot(x = cbind(-data_cluster[, 2], data_cluster[, 3]), clus = data_
    cluster[, 4]
         ,color = TRUE, lines = 0, labels = 4, plotchar = FALSE,
         main = 'Visualization of the Clusters'
         , xlab = 'Population Density', ylab = 'Secondary Industry', sub
             = NULL,
         col.p=colors_cluster)

########################################################################
# Perform Hierachical Cluster Analysis using Euclidian Distance and
# Single Linkage Fusion Algorithm

cluster_tree = hclust(dist_mat, method = "single")

# Dendrogram
plot(as.dendrogram(cluster_tree, hang = -1),main = 'Single Linkage',
     leaflab = "none")
# The dendrogram shows, that the algorithm leads to less usefull Clusters
# than Ward algorithm

########################################################################
# Perform Hierachical Cluster Analysis using Euclidian Distance and
# Centroid Algorithm
```

```
cluster_tree = hclust(dist_mat, method = "centroid")

# Dendrogram
plot(as.dendrogram(cluster_tree, hang = -1), main = 'Centroid', leaflab =
    "none")
# The dendrogram shows, that the algorithm leads to less usefull Clusters
# than Ward algorithm

##############################################################################
# Perform Hierachical Cluster Analysis using Euclidian Distance and
# Ward Algorithm for data including outliers


# Centering Variables
cent_var_outl = apply(X = as.matrix(data_cluster_outl[, 2:7]), MARGIN =
    2, FUN = scale)


# Generating modified data frame for Cluster Analysis
data_cluster_outl = cbind.data.frame(data_cluster_outl$Name, cent_var_
    outl)
data_cluster_outl = data_cluster_outl[, c("data_cluster_outl$Name", "
    PopulationDensity",
                                          "SecondaryIndustry")]
names(data_cluster_outl) = c('Name', "PopulationDensity", "
    SecondaryIndustry")
dist_mat = dist(data_cluster_outl[, c(2, 3)])

# Perform clustering algorithm
cluster_tree = hclust(dist_mat)

# Dendrogram
plot(as.dendrogram(cluster_tree, hang = -1),main = 'Dendrogram Ward
    Algorithm', leaflab = "none")

# Choose certain number of clusters according to Dendrogram
num_clus = 2
Cluster = cutree(cluster_tree, k = num_clus)

# Add Cluster Information to China Data
data_cluster_outl = cbind(data_cluster_outl, Cluster)
name_cluster      = data_cluster_outl[, c('Name', 'Cluster')]
data_china_outl   = merge(data_china, name_cluster, by = 'Name', all =
    TRUE)

# Visualize Clustering of Data
clusplot(x = cbind(-data_cluster_outl[, 2], data_cluster_outl[, 3]), clus
    =data_cluster_outl[, 4]
         ,color= TRUE, lines = 0, labels = 4, plotchar = FALSE, main = '
             Clusters if outliers included'
         ,xlab = 'Population Density', ylab = 'Secondary Industry', sub =
             NULL)

##############################################################################
# Remove all help variables
rm(gdp_percap, cent_var, Cluster, name_cluster, dist_mat, data_cluster,
    cluster_tree,
```

```
  cluster_extpop, num_clus, outlier_cities, pop_ex_max,pop_ex_min, sec_ind
    _ex_max,
  sec_ind_ex_min)

##########################################################################
# Save result in csv file
write.csv(x = data_china, file = "data_china.csv")
```

9 Appendix C

```
##########################################
##########################################
##########################################
#Extending regression by adding a variable capturing distance
#to coal power stations

#install.packages("readxl")
#install.packages("ggmap")
#install.packages("sp")
#install.packages("xtable")

library("readxl")

#library for spDists
library("sp")

#package for map
library("ggmap")

#package for latex
library("stargazer")

setwd("C:/Users/TD/Documents/Uni/Dropbox/Air Pollution")

#load data
stations = read_excel("coal.xls")
stations = stations[,c(1,4,5,7)]

#split string containing both longitude and latitude into two variables at blank space
long.lat        = as.data.frame(do.call(rbind, strsplit(stations$Coordinates, ' ')))
names(long.lat) = c("lat","long")

#Convert coordinates in degrees minutes seconds to decimal degrees

dms             = do.call(rbind, strsplit(as.character(long.lat$lat), ":"))
long.lat$lat  = as.numeric(dms[,1]) +
(as.numeric(dms[,2]) + as.numeric(dms[,3])/60)/60
rm(dms)

dms              = do.call(rbind, strsplit(as.character(long.lat$long), ":"))
long.lat$long = as.numeric(dms[,1]) +
(as.numeric(dms[,2]) + as.numeric(dms[,3])/60)/60
rm(dms)

#Load Chinese cities
data_china=read.csv("data_chinadist.csv")

#check for missing coordinates and remove rows
is.na(data_china$Latitude)
data_china=data_china[which(is.na(data_china$Latitude) == FALSE),]

#Convert into matrix format
cit  = as.matrix(cbind(data_china$Latitude,data_china$Longitude))
stat = as.matrix(long.lat)
```

```
#calculate spherical distances
dist = spDists(cit, stat, longlat=TRUE)
dist = round(dist, digits = 3)

data_china$Name = as.character(data_china$Name)
stations$Station = as.character(stations$Station)

rownames(dist) = c(data_china$Name)
colnames(dist) = c(stations$Station)
dist

#save closest station for each city and its corresponding distance
#in a variable

for (i in 1:102) {
   data_china[i,"closest"] = stations$Station[which.min(dist[i,])]
  data_china[i,"Distance"] = min(dist[i,])
}
dist
#apply Benjamins code to stations to see where they are

data_final = data_china[,c(2,3,18,19,20)]

stations = cbind(stations,long.lat)

map_aux = get_map(location = "China", maptype = "roadmap", zoom = 4,
    source = "google")
map_raw = ggmap(map_aux, base_layer = ggplot(data_final))

#map with cities
map_cities = map_raw + geom_point(color = "red", size = 1, aes(x =
    Longitude, y = Latitude))
map_cities = map_cities + labs(title = "Cities", x = "Longitude", y = "
    Latitude")
print(map_cities)

#map with cities and stations
map_citiesstations = map_cities + geom_point(data = stations, aes(x =
    long,y = lat), colour = "black", size = 1) +
  labs(title = "Cities and coal stations", x = "Longitude", y = "Latitude
    ")
print(map_citiesstations)

#now regression with distance variable
attach(data_china)
lm = lm(PM10 ~ log(GDP) + SecondaryIndustry + PopulationDensity + log(
    Imports) + (Latitude >33) + Distance, data = data_china)
summary(lm)

#latex
stargazer(lm, font.size = "small", no.space = T, intercept.bottom = T,
    single.row = T, keep.stat = c("n","adj.rsq"), digits = 2, align = T,
    report = "vc*")




#################################################################
```

```
################################################################
################################################################
#Testing of clusters

install.packages("car")
install.packages("gplots")

#for plotmeans
library("gplots")

#for Leventest and ANOVA
library("car")

setwd("C:/Users/TD/backup")
data_china = read.csv("data_chinatest.csv")

attach(data_china)

#remove NAs
data_china = data_china[is.na(data_china$Cluster) == "FALSE",]

data_china$Cluster = as.factor(data_china$Cluster)

#meansplot
plotmeans(PM10 ~ Cluster, p = 0.95, bars = T, xlab = "Cluster",
           ylab = "PM10 pollution (bars indicate 95% CI)",
           main = "Error bar diagram",
        connect = FALSE)

#normality?
tapply(data_china$PM10, data_china$Cluster, shapiro.test)

#reject for cluster 2!
plot(density(PM10[Cluster == "2"], na.rm = TRUE))

#variances equal?
leveneTest(data_china$PM10, data_china$Cluster)

leveneTest(PM10, Cluster)
##Cannot reject at 5%.


#nonparametric alternative, very conservative

mediantest  = function(x, z){
     median = median(x, na.rm = T)
     above  = (x > median)
mediantable = table(above, z)
   print(mediantable)
   chisq.test(mediantable)

}

mediantest(PM10, Cluster)

#do anova anyway
?anova
anova = aov(PM10 ~ Cluster)
summary(anova)
```

```

#post-hoc test
print(pairwise.t.test(PM10, Cluster, p.adjust.method = "bonferroni", pool.sd = TRUE))
print(pairwise.t.test(PM10, Cluster, p.adjust.method = "none"))
```

10 Appendix D

```
##########################################################################
##########   PLOTTING SPATIAL DATA USING "GGMAP"    ################

####### Clearing history and setting working directory

rm(list=ls(all=TRUE))

graphics.off()

# setwd("~/R/SPL")

####### Installing and loading required packages

install.packages("ggmap") # R version 3.3.0 is recommended

library(ggmap)

####### Reading and transforming dataset

data_china = read.csv("data_china.csv")

data_final = data_china[,c(2,3,18,19,20)]

# Assigning colours to the clusters

data_final$Colour = ifelse(is.na(data_final$Cluster) == 1, "grey", ifelse
    (data_final$Cluster == 1, "darkgreen", ifelse(data_final$Cluster == 2,
    "blue", ifelse(data_final$Cluster == 3, "red", ifelse(data_final$
    Cluster == 4, "magenta","brown")))))

attach(data_final)

head(data_final)

####### Map from Google-Maps without any additional information

map_aux = get_map(location = "China", maptype = "roadmap", zoom = 4,
    source = "google")

map_raw = ggmap(map_aux, base_layer = ggplot(data_final))

print(map_raw)

####### Map with our cities by clusters

map_cities = map_raw + geom_point(color = Colour, size = 1, aes(x =
    Longitude, y = Latitude))

map_cities = map_cities + labs(title = "Cities by Cluster", x = "
    Longitude", y = "Latitude")

print(map_cities)

####### Bubbles of the cities scaled by PM10

map_pm10 = map_raw + geom_point(aes(x = Longitude, y = Latitude, size =
    PM10), color = "red", alpha = 0.4)
```

```
map_pm10 = map_pm10 + labs(title = "Air Pollution in Chinese Cities", x =
    "Longitude", y = "Latitude", size = "PM10 ug/m3")

print(map_pm10)

####### Shanghai area marked with a blue rectangle

# size of the box: [longitude: 116-124, latitude: 28-34]

rect = data.frame(xmin = 116, xmax = 124, ymin = 28, ymax = 34)

map_rect = map_pm10 + geom_rect(data = rect, aes(xmin = 116, xmax = 124,
    ymin = 28, ymax = 34), color = "gray20", alpha = 0.4, inherit.aes =
    FALSE)

print(map_rect)

####### Zooming into the map (only in the presentation slides)

map_aux2 = get_map(location = c(lon = 120, lat = 31), maptype = "roadmap"
    , zoom = 7, source = "google")

map_zoom = ggmap(map_aux2, base_layer = ggplot(data_final, aes(x =
    Longitude, y = Latitude)))

map_zoom = map_zoom + geom_point(aes(size = PM10), color = "red", alpha =
    0.5) + labs(title = "Air Pollution - Shanghai Area", x = "Longitude",
    y = "Latitude", size = "PM10 ug/m3")

print(map_zoom)

###########################################################################
###########################################################################
```