Statistical Analysis and Evaluation of Air Quality in an Italiancity from 2004 to 2005

Yiziying(Kimmi) Chen

Introductions:

In our final project, we used a dataset about air quality (recorded from 2004 to 2005) to analyze distributions of different variables with methods we learned in this class. The full dataset contains 9358 observations and 15 variables. All observations are the collection of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. This device was located in one severely polluted area in an Italian city. To assist our analysis, we use the distribution provided in the article, Monte Carlo uncertainty analysis of a regional-scale transport chemistry model constrained by measurements from the Atmospheric Pollution Over the Paris Area (ESQUIF) campaign (Beekmann et al,. 2003), which is the uncertainty densitydistributiononatmosphericchemistrymodels.

Datasetdescription:

Variables	Description
Date	day/month/year
Time	hour/minute/second
CO	Truehourlyaverageconcentrationinmg/m^3
PT08.S1(CO)	tinoxide,hourlyaveragedsensorresponse(nominallyCOtargeted)
NMHC(GT)	TruehourlyaveragedoverallNonMetanicHydroCarbonsconcentrationin microg/m^3
C6H6(GT)	TruehourlyaveragedBenzeneconcentrationinmicrog/m^3
PT08.S2(NMHC)	Titania,hourlyaveragedsensorresponse(nominallyNMHCtargeted)
NOx(GT)	TruehourlyaveragedNOxconcentrationinppb
PT08.S3(NOx)	tungstenoxide,hourlyaveragedsensorresponse(nominallyNOxtargeted)
NO2(GT)	TruehourlyaveragedNO2concentrationinmicrog/m^3
PT08.S4(NO2)	tungstenoxidehourlyaveragedsensorresponse(nominallyNO2targeted)
PT08.S5(O3)	indiumoxide,hourlyaveragedsensorresponse(nominallyO3targeted)
T	Temperature
RH	RelativeHumidity
AH	AbsoluteHumidity

We first cleaned the dataset and left 827 observations with no missing values. Our project's main goal is to apply the methods learnt from this course and analyze which chemical oxide makes heaviest effect on air pollution issues. Ratio statistic between air pollutant and its targeting metal oxide was taken into consideration for interpreting the amount of air pollutant present in the air regarding the unit metal oxide detected by the sensor. Besides that, we would like to analyze the relationship among air pollutant variables. We included correlation statistics and also wanted to

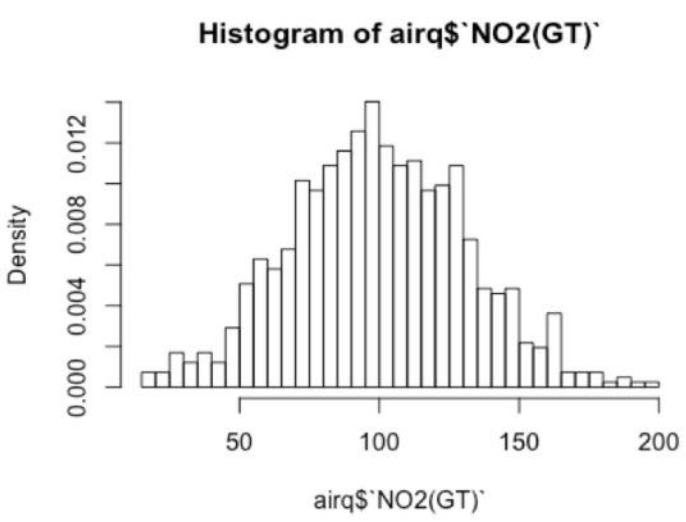
check for similar distribution between variables. To achieve our goal, we used 10 methods (including random number generation embedded in all other 9 methods) regarding different statistical analysis to help achieving the central goal proposed for this project. Based on the statistical results, we will give advice to the possible solutions for treating air quality issues. The methods utilized in this project can be further applied to other envoironment related datasets in different areas/cities around the world, thereby assisting specialists to improve environmental quality. In the following parts, we will further illustrate how we use these methods to obtain the desiredresultsandgivedetailedanalysistothem.

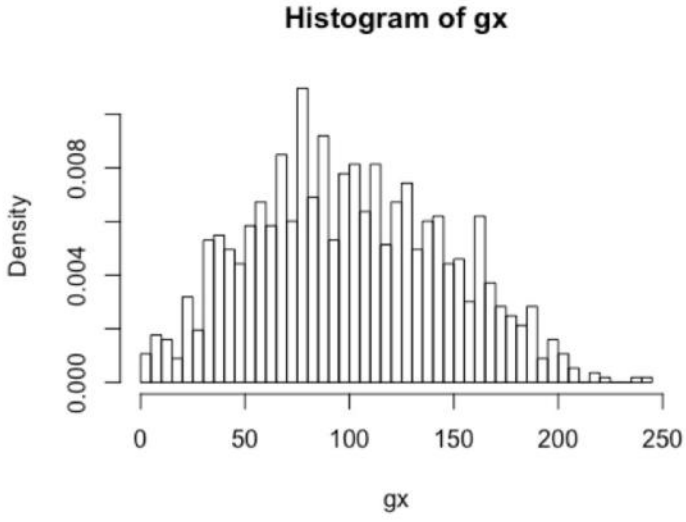
Method and Result:

First, we read the data and eliminate the missing values using the codes **Data input and cleaning** fromappendix.

1. Accept-Reject

Before we use the density function acquired from the article: $p(O|Y_k) = \frac{1}{\sqrt{2\pi}}\frac{1}{\sigma_\varepsilon} exp(-0.5[(O - Y_k)/\sigma_\varepsilon]^2)$, we would like to check whether this function can predict the distribution of the observation in the dataset correctly. To identify if this function works for the dataset, we generate some observations from the function. After that, we draw distribution graphs of the samples we generated from the function as well as from the observations provided by the dataset. Comparing these two groups of graphs, we can check if the function fit the dataset well. To generate random variables, we need means, variance and a similar distribution function which has thicker tails. Means and standard deviations are generated by observations of each variable from the dataset. Additionally, we choose the exponential distribution and uniform distribution as candidate distributions, while uniform distribution does not turn out well. As a result, we keep the exponential distribution. The sample code we used to generate those graphs can be inferred from appendix:**Accept-Reject**section.





In the result, only a few variables do not follow the distribution function (from article) and we feel confidenttousethisfunctioninanalyzingmostofvariables.

2.Variance Reduction

Since data follows normal distribution, we'd like to reduce its variance. As the distribution is notmonotone within its range, we used the stratified sampling to perform variance reduction. The samplecode can be inferred from the appendix: **Variance Reduction** section.

3Monte-CarloInference

In statistical inference, there is uncertainty in an estimation process. The Monte-Carlo inference method repeats sampling from a given probability model to investigate this uncertainty. In the analysis of air quality, we use Monte-Carlo inference to compute the empirical confidence level of variance for air pollutants obtained by simulation and assess type I error rate of variance for air pollutants.

First, let's focus on the MC estimate of confidence level of the variance for C6H6. We assume that it follows normal distribution. If $X1, \ldots, Xn$ is a random sample from a $N(\mu, \sigma^2)$ distribution, $n \geq 2$, and S^2 isthesamplevariance,then $V = \frac{(n-1)S_2}{\sigma^2} \sim \chi^2(n-1)$

CI	StandardError
0.955	0.006556

The result shows that 955 intervals satisfied $U\ CL > \sigma^2$, so the empirical confidence level is 95.5% in this experiment. The result will vary but should be close to the theoretical value, 95%. The standarderroroftheestimateis0.006556.

Next, let's focus on the MC estimate of type I error of the variance for C6H6, we assume that it follows normal distribution. Suppose that $X_1, \ldots, X_n$ is a random sample from $N(\mu, \sigma)$ distribution. Test$H_0 : \mu_1$ vs$H_1 : \mu > \mu_1$ at$\alpha = 0.05$.

Underthenullhypothesis,$T = \frac{\bar{x} - \mu_1}{S/\sqrt{n}} \sim t(n-1)$

TypeIerror	StandardError
0.0501	0.002073

The observed Type I error in this simulation is 0.0501 and the standard error of the estimate is approximately 0.0021. Estimates of Type I error probability will vary, but should be close to the nominal rate = 0.05 because all samples were generated under the null hypothesis from the assumedmodelforat-test(normaldistribution).

Note: the code used to generate the statistical results can be found at appendix: **MC Inference** section.

4Bootstrap

We are also interested in determining the density distributions of several air pollutants population. The two main statistics we focused on are correlation and ratio (air pollutant:targeting metal oxide). In order to estimate the population distributions, for each variable, we utilize bootstrap to generate random samples (by resampling) from its empirical distribution and estimate population

characteristics and make inferences about the sampled population. To obtain an estimate of the shape of density, histograms will be generated. We utilize the real correlation and real ratio as the unbiasedestimatorsforobtainingtheestimatedbias.

Using an unbiased estimator, the bias of an estimator can be obtained: $\text{bias}(\hat{\theta}) = E[\hat{\theta} - \theta] = E[\hat{\theta}] - \theta$. The bootstrap estimation of bias uses the bootstrap replicates of an estimator to estimate the sampling distribution of the estimator: $\widehat{bias}(\hat{\theta}) = \bar{\theta}^{*} - \hat{\theta}$,where $\bar{\theta}^{*} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^{*(b)}$, and $\hat{\theta} = \hat{\theta}(x)$ is the estimate computed from the original observed sample. The code used to generate the correlation results and histograms can be inferred from appendix: **Bootstrap-CorrelationStatistic**section.

Bootstrapcorrelationsummary

	Real correlation	Estimated Correlation	Estimated standard error	Estimated Bias
C6H6& NMHC	0.8979	0.8978	0.00678	-0.0001
CO& C6H6	0.9726	0.9621	0.00227	-0.0004
CO& NMHC	0.8871	0.8866	0.00819	-0.0005
NO2& NOx	0.8574	0.8579	0.00972	0.0005

From the correlation summary table, we see that CO and C6H6 has the highest correlation value, lowest estimated standard error and bias. It is possible that these two pollutants were emitted simultaneously during an industrial processing step. By acquiring long-term air quality data records and the related information about industry production procedures, we may draw conclusions to the detailed reasons of this high correlation value. By acquiring additional information to the city's industry processing methods, we may give advice to improve air quality utilizing the correlation informationbetweenmainairpollutants.

BootstrapRatioSummary

	Real Ratio	Estimated Ratio	Estimated standard error	Estimated Bias	Cross-Validation
CO/tin oxide	0.001948	0.00195	2.875e-5	2.083e-6	0.0724
NMHC/ titania	0.2391	0.2396	0.00046	0.0051	0.0904
NOx /tungsten	0.1489	0.1482	-0.00072	0.0043	-0.1676
NO2/ tungsten	0.0626	0.0626	-3.3804e-5	0.0004	-0.0760

From the ratio summary, we can interpret the amount of metal oxide and its relating air pollutant amount in the air. For example, for 1 unit of titania detected by the sensor, there is about 0.24 unit of NMHC pollutant presented in the air; for 1 unit of tungsten detected by the sensor, there is about 0.15 unit of NOx pollutant presented in the air. Using these statistical results, we can easily target the highest ratio of air pollutant to their targeting metal oxide. These results can help to offer

applicable advice in targeting some high-level air pollutants and suggesting useful solutions to optimize metal oxidation/reduction processes which will further reduce relevant pollutants' amount present in the air. The code used to generate the correlation results and histograms can be inferredfromappendix:**Bootstrap-RatioStatistic**section.

5Jackknife

From the previous part using Bootstrap method, we found the highest correlation belongs to CO and C6H6. Following the bootstrap method, we try another resampling method, Jackknife, to compute the this high correlation between CO and C6H6. We obtain the estimated bias and standarderrorintheresults.

A Jackknife estimate of standard error is: $\hat{se}_{jack} = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}_{(i)} - \hat{\theta}_{(.)})^2}$. A factor of $(n-1)\n$ undertheradialmakes$\hat{se}_{jack}$ anunbiasedestimatorofthestandarderrorofthemean.

SummaryofJackkniferesamplingcor(CO,C6H6):

Jackknife bias	JackknifeStandard error	Realcor(CO, C6H6)
2.936e-06	0.002301	0.9727

From the result table, we can find the bias and standard error are both small, and the correlation between CO and C6H6 is above 0.97. The statistical measurements from Jackknife are close to the results using Bootstrap method. The code used in Jackknife can be found at appendix: **Jackknife** section.

6JackknifeafterBootstrap

Then, we do Jackknife-after-Bootstrap to see whether we can reduce the standard error of correlation between CO and C6H6. The bootstrap estimate of standard error is 0.0023 and jackknife-after-bootstrap estimate of its standard error is 0.00463.We compare the standard errors fromjackknife,bootstrapandjackknife-after-bootstrapmethodsasbelow:

Method	Standarderror
Jackknife	0.002301
Bootstrap	0.002457
Jackknifeafter Bootstrap	0.004631

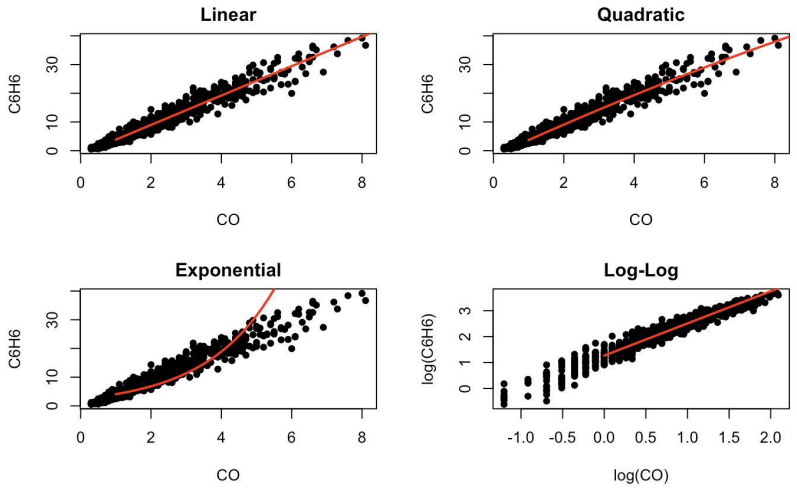
We find that Jackknife is the best with smallest standard error. Jackknife after bootstrap is no better than previous two methods. Thus, Jackknife is the best resampling method among 3 in computing thecorrelationbetweenCOandC6H6whileloweringestimatedstandarderrorusingourdataset.
The code used in generate the statistical results can be found at appendix: **Jackknife after Bootstrap**section.

7Application:CrossValidation

Then we do the research about the relation between CO and C6H6 using cross validation. We choose4differentmodelstofitthedata:

1. Linear: $Y = \beta_0 + \beta_1 X + e$.
2. Quadratic: $Y = \beta_0 + \beta_1 X + \beta_1 X^2 + e$.
3. Exponential: $log(Y) = log(\beta_0) + \beta_1 X + e$.
4. Log-Log: $log(Y) = \beta_0 + \beta_1 log(X) + e$.

In order to choose the best model, we estimate prediction error by n-fold (leave-one-out) cross validation.



From the model plots, it is hard to draw a conclusion that which model performs the best in fitting our data. Cearly, model seems to fit the worst. In order to select out the best model, we generate predictionerrorformodelcomparison.
Thefollowingestimatesforpredictionerrorareobtainedfromthen-foldcrossvalidation:

e1	e2	e3	e4
2789	1488	3.837e+15	6675

According to the prediction error criterion, Model 2, the quadratic model, would be the best fit for ourdataasithasthelowesterrorcomparedwithusingtheotherthreemodels.
SummarytableforModel2:

Molde-295%ConfidenceInterval	2.5%	97.5%
(intercept)	-2.2199	-1.46825
CO	5.3427	5.89277
I(C6H6^2)	-0.1227	-0.03819

```
Call:
lm(formula = C6H6 ~ CO + I(CO^2))
Residuals:
Min 1Q Median 3Q Max
-9.017 -0.982 -0.063 0.874 6.579
Coefficients:
Estimate Std. Error t value Pr(>|t|)
```

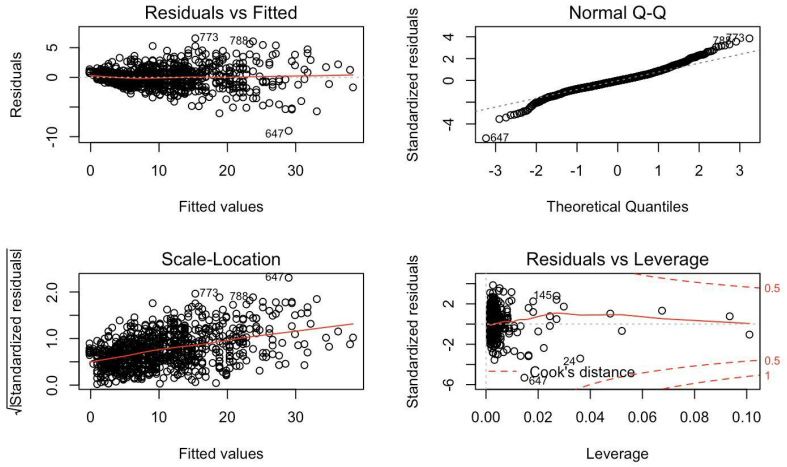
```
   (Intercept) -1.8441 0.1915 -9.63 <2e-16 *** CO 5
   .6177 0.1401 40.09 <2e-16 *** I(C6H6^2) -0.0804
                          0.0215 -3.74 0.0002 ***
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.71 on 824 degrees of freedom

Multiple R-squared: 0.947, Adjusted R-squared: 0.947

F-statistic: 7.36e+03 on 2 and 824 DF, p-value: <2e-16
```

The fitted regression equation for Model 2 is $Y = -1.8441 + 5.6177x - 0.0804x^2$. From the result, we find that all predictors' coefficients have p values smaller than 0.05, meaning they are all significant predictors.Andwecangetthe95%confidenceintervalofthecoefficients.



From the diagnostic plots, we see that the residual are normal distributed from the right top plot andtheleftbottomplotshowsthattheresidualdonothaveconstantvariance.

Note: the code used for generating the statistical results and plots can be inferred from appendix: **CrossValidation**section.

8PermutationTests

Datafeatures:

Airpollutant	Meanvalue
NOx	143.5018
NO2	100.2599
CO	2.3535
NMHC	231.0254
C6H6	10.7723

Airpollutant	Range
NOx	12-478

NO2	19-196

Airpollutant	Meanvalue
S1-CO	1207.7418
S2-NMHC	965.9837
S3-NOx	963.1780
S4-NO2	1600.5065
S5-O3	1045.6910

Airpollutant	Range
S2-NMHC	447.5-1754.25
S3-NOx	461.25-1934.5

From the above outputs, we see S2-NMHC and S3-NOx seems to have similar mean values and similar ranges. So we address a question, whether S2-NMHC share same distribution with S3-NOx? How about NOx and NO2? Do they share similar distribution? In order to get the answer, we conductpermutationtestusing2-wayt-testandKolmogorov-Smirnovtest.

PermutationtestforS2-NMHCandS3-NO:

Test	pvalue
Ttest	0.42
K-Stest	0.152

The p-value from two tests for S2-NMHC & S3-NOx are both >0.05, so we accept the null hypothesis andconcludethesetwovariablessharesimilardistribution.

PermutationtestforNOxandNO2:

Test	pvalue
Ttest	0.001
K-Stest	0.001

OppositeconclusionisgiventoNOx&NO2sincetheirtestp-valuesareboth<0.05.

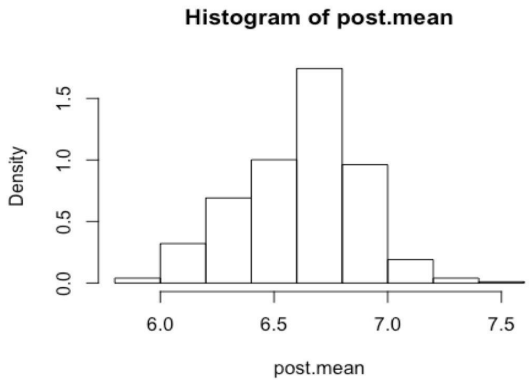
9BayesianInference&Monte-CarloIntegration

Bayesian Inference is based on the Bayesian Theorem. The biggest difference of Bayesian approach is that it views the unknown parameters as random variables. Based on that we can estimate the unknown parameter by computing the posterior means, posterior modes and so on. Because the variables we are using are all continuous, we will refer to the continuous form, which is as following:$f_{X|Y=y}(x) = \frac{f_{Y|X=x}(y) f_X(x)}{f_Y(y)} = \frac{f_{Y|X=x}(y) f_X(x)}{\int_{-\infty}^{\infty} f_{Y|X=x}(y) f_X(x) dx}$

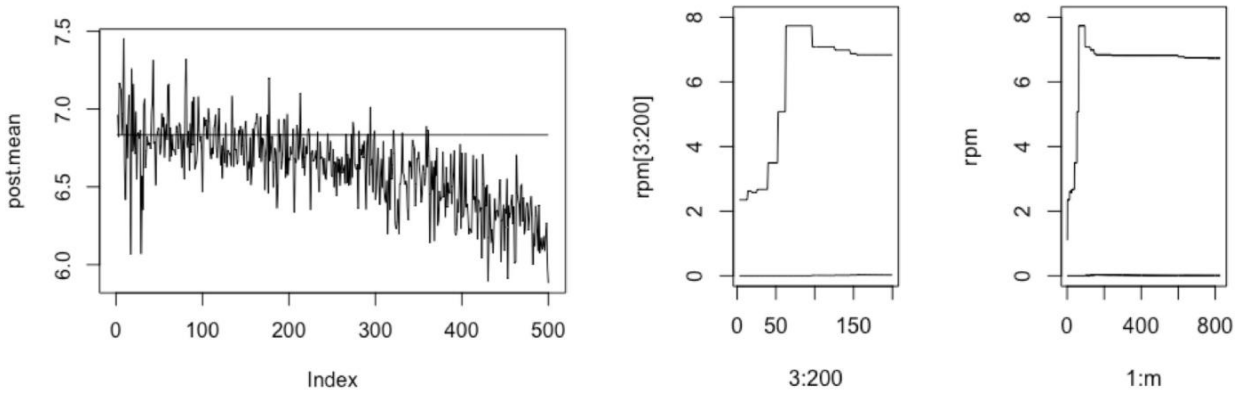
Suppose we assume X has the density $f(x|\theta)$, then the posterior density of θ can be reached by the equationas follows:$f_{\theta|x}(\theta) = \frac{f(x_1,\ldots,x_n|\theta) f_\theta(\theta)}{\int f(x_1,\ldots,x_n|\theta) f_\theta(\theta) d\theta}$

Usually, the integration in the denominator is difficult to calculate and it is where we can apply Monte-Carlomethod.

Here, for the application purpose, we will take the variable PT08.S3(NOx) as an example. First, we checkthehistogramshownasfollowing:



As we can see, the histogram shows a very good symmetric shape and we can assume that our variable roughly obeys a normal distribution $X \sim N(\mu, \sigma)$. Then we do a Bayesian Inference and estimatetheposteriormeanwhichistheμ wewanttoget.



From the plots, we can see that the two lines become stable at around 7 and we can draw the conclusion that posterior mean value converges. Also from the summary table below, we see the posterior mean is close to sample mean and is a little below 7, matching with the convergence resultontheplots.

NormalizingconstantC	Posteriormean	Samplemean
0.01012	6.732	6.834

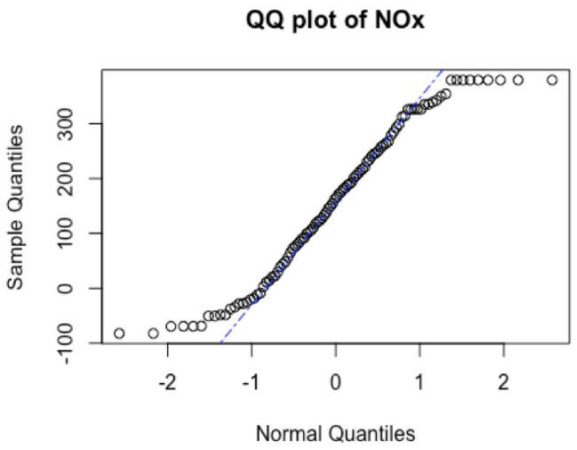
Next, we adjust the *sigma* to get a range with good posterior means. As a result, we get information from the graphs that when we make 2% disturb on the σ , the posterior means are around the sample mean value and the histogram is symmetric. (Note: the code used for generating the statistical results and graphs can be found at appendix: **Bayesian Inference & Monte-Carlo Integration**section.)

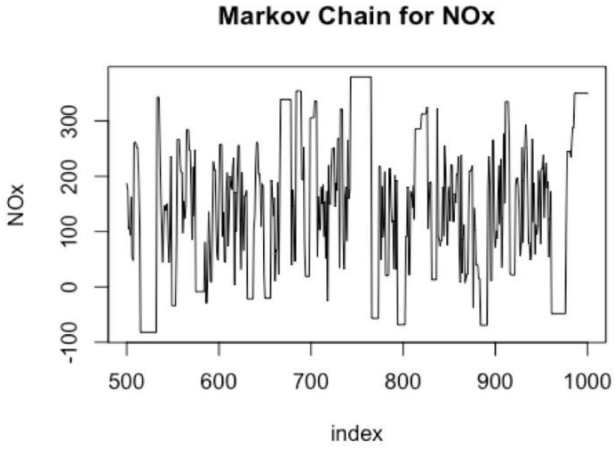
10MCMC

Metropolis-Hastingssampler

To test the efficiency of applying MCMC to our dataset, we picked Metropolis-Hastings sampler and RandomWalkMetropolisusingtargetdistribution:$p(O|Y_k) = \frac{1}{\sqrt{2\pi}\sigma_\varepsilon} exp(-0.5[(O - Y_k)/\sigma_\varepsilon]^2)$

Where Y_k is from proposed normal distribution, with observation mean and observation standard deviation.Forexample,weuseNOxvariabletodrawMarkovChainplot.





AcceptancerateforNOx	AcceptancerateforNO2	AcceptancerateforCO
53%	28.4%	27.3%

The above chain plot does not present the markov chain very well as we can see some horizontal segments. The acceptance rate is about 53%, which is not very high but acceptable. The QQ plot gives a visual goodness of fit test, suggesting that NOx roughly fit a normal distribution. The code used for generating the results can be inferred from appendix: **MCMC-Metropolis-Hastings sampler**section.

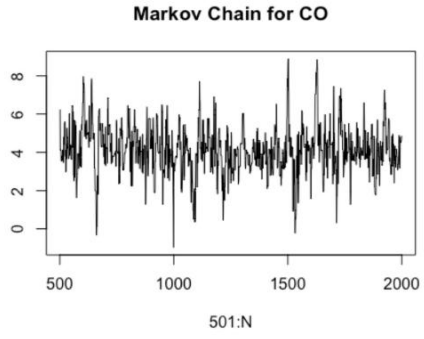
RandomWalkMetropolis

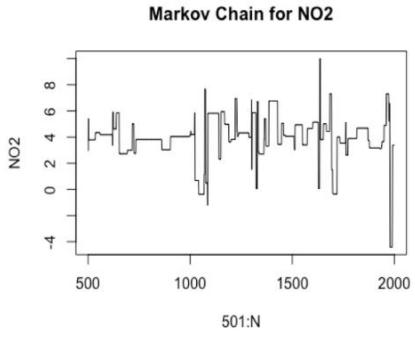
Acceptanceresults:

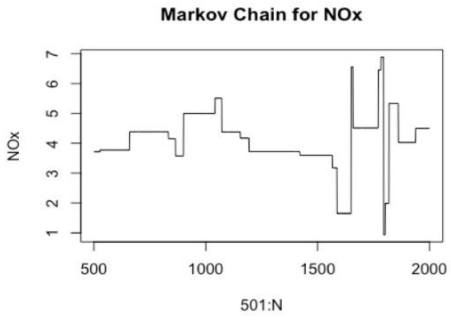
AcceptancerateforNO2	AcceptancerateforNOx	AcceptancerateforCO
0.059	0.023	0.617

Variance foreachvariable:

VarianceofNO2	VarianceofNOx	VarianceofCO
991.8609	6696.103	1.9866







From the acceptance rate results, we see CO has the highest acceptance rate as 61.7%. Also, CO has the best chain plot as there is no obvious horizontal line, and the chain plot roughly converges at 4.

Random Walk Metropolis's convergence is sensitive to the scalar parameter and because CO has the smallest variance, it is less likely to reject its candidate points and has the best acceptance rate. In the future data analysis projects, we will pay more attention to observation variances that could affect the efficiency of markov chain. The code used for generating the results can be inferred from appendix:**MCMC-RandomWalkMetropolis**section.

Conclusion:

By applying the methods above, we can get a statistical vision of the data. First, most variables or logarithm of variables, such as NO2, PT08.S1(CO) and so on, are of bell-shaped histogram indicating they roughly obey normal distribution or lognormal distribution. Second, assuming variable PT08.S3(NOx) is of lognormal distribution, we estimate the posterior mean which is around the sample mean and get the adjusted range of σ is around 2%. Third, using bootstrap method, we find that CO and C6H6 has the highest correlation and it is possible that these two air pollutants are simultaneously emitted to the air during industrial process. Therefore, when targeting these air pollutants, we can focus on reducing these CO and C6H6 at the same time. Finally, from the permutation tests, we see that S2 Titania and S3 tungsten oxide share similar distribution with close center (mean). The industry companies in that Italian city might utilize titania and tungsten in similar industrial materials. If we can obtain the data for the production of titania and tungsten and explore their similar functions in industrial procedures, we may use the same method to optimize titania and tungsten oxide smelting process to reduce air pollutants. In all, CO is the most significant determinantofairqualityinthisdataset.

Reference:

1. Beekmann, Matthias, and Claude Derognat. "Monte Carlo uncertainty analysis of a regional scale transport chemistry model constrained by measurements from the Atmospheric Pollution Over the Paris Area (ESQUIF) campaign." Journal of Geophysical Research: Atmospheres,7Aug.2003

2. S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia, On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario, Sensors and Actuators B:Chemical,Volume129,Issue2,22February2008,Pages750-757,ISSN0925-4005

Appendix:

I.Codeusedingeneratingresults:

```
options(scipen = 1, digits = 4, width = 80)
library(knitr)
opts_chunk$set(cache=TRUE, autodep=TRUE, cache.comments=FALSE,
 message=FALSE, warning=FALSE, results = "hide",eval=FALSE)
```

Datainputandcleaning

```
library(openxlsx)
AIR = read.xlsx('AirQualityUCI.xlsx')
airq= AIR[AIR$`CO(GT)` != -200 & AIR$`PT08.S1(CO)` != -200 & AIR$`NMHC(GT)` != -200 &
AIR$`C6H6(GT)` != -200 & AIR$`PT08.S2(NMHC)` != -200 & AIR$`NOx(GT)` != -200 &
```

```
AIR$`NO2(GT)` != -200 & AIR$`PT08.S3(NOx)` != -200 & AIR$`PT08.S4(NO2)` != -200 & AIR$
`PT08.S5(O3)` != -200 & AIR$T != -200 & AIR$RH != -200 & AIR$AH != -200,]
```

Accept-Reject

```
meanno2=mean(airq$`NO2(GT)`)
sdno2=sd(airq$`NO2(GT)`)
n = 10000
Ykno2=rnorm(n,meanno2,sdno2)
gx = rexp(n,1/sdno2^2)
for(i in 1:n)
gx[i]=gx[i]*(runif(1)<exp(-(((gx[i]-Ykno2[i])^2)/(2*sdno2))+abs(gx[i]-Ykno2[i])-1/(2
*sdno2)-1))
gx=gx[gx!=0]
hist(airq$`NO2(GT)`,freq=F,breaks=50)
hist(gx,freq=F,breaks=50)
```

VarianceReduction

```
M <- 20 ;T2 <- numeric(4);estimates <- matrix(0, 10, 2)
f = function(x) {
sigma = sdno2
Ykno2=rnorm(n,meanno2,sdno2)
return(exp(-0.5*(x-Ykno2)/sigma)^2/(sqrt(2*pi)*sigma))} for (i in 1:
10) {
estimates[i, 1] <- mean(f(runif(M)))
T2[1] <- mean(f(runif(M/4, 0, .25)))
T2[2] <- mean(f(runif(M/4, .25, .5)))
T2[3] <- mean(f(runif(M/4, .5, .75)))
T2[4] <- mean(f(runif(M/4, .75, 1)))
estimates[i, 2] <- mean(T2)
apply(estimates, 2, mean)
apply(estimates, 2, var)}
```

MCInference

(1)MCestimateofCI

```
set.seed(000)
CO=airq$`CO(GT)` ;C6H6=airq$`C6H6(GT)`
mu1=mean(CO);mu2=mean(C6H6)
sd1=sd(CO);sd2=sd(C6H6)
n <- 20;alpha <- .05
UCL <- replicate(1000, expr = {
x <- rnorm(n, mean = mu2, sd = sd2)
(n-1) * var(x) / qchisq(alpha, df = n-1) })
#count the number of intervals that contain sd2^2 Num=sum(UCL >
sd2^2)
#or compute the mean to get the confidence level
CI=mean(UCL > sd2^2)
p.hat=1-CI
se <- sqrt(p.hat * (1 - p.hat) / 1000)
cbind(CI,se)
```

(2)MCestimateofTypeIerror

```
set.seed(000)
n <- 20 ;alpha <- .05 ;mu0 <- mu2;sigma <- sd2;  m <- 10000
#number of replicates
p <- numeric(m) #storage for p-values
for (j in 1:m) {
x <- rnorm(n, mu0, sigma)
ttest <- t.test(x, alternative = "greater", mu = mu0)
p[j] <- ttest$p.value
}
type_1_error <- mean(p < alpha)
se <- sqrt(p.hat * (1 - p.hat) / m)
cbind(type_1_error, se)
```

BootstrapCorrelationStatistic

(1) CorrelationforC6H6&NMHC

```
library(bootstrap)
c6h6nmhc.hat = (cor(airq$`C6H6(GT)`,airq$`NMHC(GT)`))
B <- 200 #number of replicates
n <- length(airq$`C6H6(GT)`) #sample size
R <- numeric(B) #storage for replicates
for (b in 1:B) {
 i <- sample(1:n, size = n, replace = TRUE)
 C6H6 <- airq$`C6H6(GT)`[i]
 NMHC <- airq$`NMHC(GT)`[i]
 R[b] <- cor(C6H6, NMHC)
}
par(ps=8)
cbind(real.corr = c6h6nmhc.hat, corr.est = mean(R), Sd.est = (se.R <- sd(R)),
Bias.est = mean(R-c6h6nmhc.hat))
hist(R, prob = TRUE, breaks = 30, xlab = "Correlation between C6H6 and NMHC", main =
"Histogram of Bootstrap estimation for correlation between C6H6&NMHC")
```

(2) CorrelationforCO&C6H6

```
set.seed(000)
coc6h6.hat = cor(airq$`CO(GT)`,airq$`C6H6(GT)`)
B <- 200 #number of replicates
n <- length(airq$`CO(GT)`) #sample size
R <- numeric(B) #storage for replicates


for (b in 1:B) {
 i <- sample(1:n, size = n, replace = TRUE)
 CO <- airq$`CO(GT)`[i]
 C6H6 <- airq$`C6H6(GT)`[i]
 R[b] <- cor(CO,C6H6)
}

cbind(real.corr = coc6h6.hat,corr.est = mean(R),Se.est = (se.R <- sd(R)), Bias.est =
```

```
mean(R-coc6h6.hat))
par(ps=10)
hist(R, prob = TRUE, breaks = 30, xlab = "Correlation between CO and C6H6",main =
"Histogram of Bootstrap estimation for correlation between CO&C6H6")
```

(3) CorrelationforCO&NMHC

```
conmhc.hat = cor(airq$`CO(GT)`,airq$`NMHC(GT)`)
B <- 200 #number of replicates
n <- length(airq$`CO(GT)`) #sample size
R <- numeric(B) #storage for replicates


for (b in 1:B) {
 i <- sample(1:n, size = n, replace = TRUE)
 CO <- airq$`CO(GT)`[i]
 NMHC <- airq$`NMHC(GT)`[i]
 R[b] <- cor(CO,NMHC)
}

cbind(real.corr = conmhc.hat,corr.est = mean(R),Se.est = (se.R <- sd(R)), Bias.est =
mean(R-conmhc.hat))
par(ps=10)
hist(R, prob = TRUE, breaks = 30, xlab = "Correlation between CO and NMHC", main =
"Histogram of Bootstrap estimation for correlation between CO&NMHC")
```

(4) CorrelationforNOx&NO2

```
noxno2.hat = cor(airq$`NOx(GT)`,airq$`NO2(GT)`)
B <- 200 #number of replicates
n <- length(airq$`NOx(GT)`) #sample size
R <- numeric(B) #storage for replicates


for (b in 1:B) {
 i <- sample(1:n, size = n, replace = TRUE)
 NOx <- airq$`NOx(GT)`[i]
 NO2 <- airq$`NO2(GT)`[i]
 R[b] <- cor(NOx, NO2)
}

cbind(real.corr =noxno2.hat, corr.est = mean(R),Se.est = (se.R <- sd(R)), Bias.est =
mean(R-noxno2.hat))
par(ps=10)
hist(R, prob = TRUE, breaks = 30, xlab = "Correlation between NOx and NO2", main =
"Histogram of Bootstrap estimation for correlation between NOx&NO2")
```

BootstrapRatiostatistic

```
B <- 1000
theta.b <- numeric(B)
```

(1) RatioEstimateforCOtotinoxide

```
theta.co <- mean(airq$`CO(GT)`) / mean(airq$`PT08.S1(CO)`)
cbind(real.ratio = theta.co)
B <- 200 #number of replicates
n <- nrow(airq) #sample size
R1 <- numeric(B) #storage for replicates
for (b in 1:B) {
 i <- sample(1:n, size = n, replace = TRUE)
 CO <- airq$`CO(GT)`[i]
 S1_CO <- airq$`PT08.S1(CO)`[i]
 R1[b] <- mean(CO)/mean(S1_CO)
}
bias1 <- mean(R1) - theta.co
se1 <- sd(R1)
cbind(real.ratio = theta.co,ratio.est = mean(R1),Se.est = se1, Bias.est = bias1, CV = bias1/se1)
par(ps=10)
hist(R1, prob = TRUE, breaks = 30, xlab = "Chemical Equivalence of CO", main =
"Histogram of Bootstrap estimation of Chemical Equivalence of CO")
```

(2) RatioEstimateforNMHCtotitania

```
theta.nmhc <- mean(airq$`NMHC(GT)`) / mean(airq$`PT08.S2(NMHC)`)
R2 <- numeric(B)
#bootstrap
for (b in 1:B) {
i <- sample(1:n, size = n, replace = TRUE)
NMHC <- airq$`NMHC(GT)`[i]
S2 <- airq$`PT08.S2(NMHC)`[i]
R2[b] <- mean(NMHC) / mean(S2)
}
bias2 <- mean(R2) - theta.nmhc
se2 <- sd(R2)
cbind(real.ratio=theta.nmhc, ratio.est = mean(R2),Bias.est = bias2, Se.est = se2, CV = bias2/
se2)
par(ps=10)
hist(R2, prob = TRUE, breaks = 30, xlab = "Chemical Equivalence of NMHC",main =
"Histogram of Bootstrap estimation of Chemical Equivalence of NMHC")
```

(3) RatioEstimateforNOxtotungsten

```
theta.nox <- mean(airq$`NOx(GT)`) / mean(airq$`PT08.S3(NOx)`)
#bootstrap
R3 <- numeric(B)
for (b in 1:B) {
i <- sample(1:n, size = n, replace = TRUE)
NOx <- airq$`NOx(GT)` [i]
S3 <- airq$`PT08.S3(NOx)` [i]
```

```
R3[b] <- mean(NOx) / mean(S3)
}
bias3 <- mean(R3) - theta.nox
se3 <- sd(R3)
cbind(real.corr=theta.nox, ratio.est = mean(R3),bias.est = bias3, Se.est = se3, CV =
bias3/se3)
par(ps=10)
hist(R3, prob = TRUE, breaks = 30, xlab = "Chemical Equivalence of NOx",main =
"Histogram of Bootstrap estimation of Chemical Eioequivalence of NOx")
```

(4) RatioEstimateforNO2totungsten

```
theta.no2 <- mean(airq$`NO2`) / mean(airq$`PT08.S4(NO2)`)
#bootstrap
R4 = numeric(B)
for (b in 1:B) {
i <- sample(1:n, size = n, replace = TRUE)
NO2 <- airq$`NO2` [i]
S4 <- airq$`PT08.S4(NO2)` [i]
R4[b] <- mean(NO2) / mean(S4)
}
bias4 <- mean(R4) - theta.no2
se4 <- sd(R4)
cbind(real.ratio=theta.no2, ratio.est = mean(R4),Bias.est = bias4, Se.est = se4, CV = bias4/se4)
par(ps=10)
hist(R4, prob = TRUE, breaks = 30, xlab = "Chemical Equivalence of NO2",main =
"Histogram of Bootstrap estimation of Chemical Equivalence of NO2")
```

Jackknife

```
set.seed(000)
CO=airq$`CO(GT)`
C6H6=airq$`C6H6(GT)`
n=length(CO)
(real.corr=cor(CO,C6H6))
corr.est=numeric(n)
for(i in 1:n){
 corr.est[i]=cor(CO[-i],C6H6[-i])
}
biasjack=(n-1)*(mean(corr.est)-real.corr)
sumsq=sum((corr.est-mean(corr.est))^2)
sejack=sqrt((n-1)/n)*sqrt(sumsq)
cbind(biasjack,sejack)
par(ps=7)
hist(corr.est,xlab='correlation between CO and C6H6',breaks = 40,main='Histogram of
Jackknife estimation for correlation between CO&C6H6')
```

JackknifeafterBootstrap

```
set.seed(000)
CO=airq$`CO(GT)`
C6H6=airq$`C6H6(GT)`
#initialize
```

```r
n <- length(CO);B <- 200;theta.b <- numeric(B);indices <- matrix(0, nrow = B, ncol = n)
# jackknife-after-bootstrap step 1: run the bootstrap
for (b in 1:B) {
i <- sample(1:n, size = n, replace = TRUE)
x <- CO[i];z <- C6H6[i];theta.b[b] <- cor(x,z) #save
the indices for the jackknife indices[b, ] <- i
}
#jackknife-after-bootstrap to est. se(se)
se.jack <- numeric(n)
for (i in 1:n) {
#in i-th replicate omit all samples with x[i]
 keep <- (1:B)[apply(indices, MARGIN = 1, FUN =
function(k) {!any(k == i)})]  se.jack[i] <- sd
(theta.b[keep])
}
(bootstrap_sd=sd(theta.b))
(jackknife_bootstrap_sd=sqrt((n-1) * mean((se.jack - mean(se.jack))^2)))
```

CrossValidationApplications

```r
CO=airq$`CO(GT)`
C6H6=airq$`C6H6(GT)`
a <- seq(1,10,0.1) #sequence for plotting fits
L1 <- lm(C6H6 ~ CO)
par(mfrow=c(2,2))
plot(CO, C6H6, main="Linear", pch=16)
yhat1 <- L1$coef[1] + L1$coef[2]*a
lines(a, yhat1, lwd=2,col='red')
L2 <- lm(C6H6 ~ CO + I(CO^2))
plot(CO, C6H6, main="Quadratic", pch=16)
yhat2 <- L2$coef[1] + L2$coef[2]*a + L2$coef[3]*a^2
lines(a, yhat2, lwd=2,col='red')
L3 <- lm(log(C6H6) ~ CO)
plot(CO, C6H6, main="Exponential", pch=16)
logyhat3 <- L3$coef[1] + L3$coef[2]*a
yhat3 <- exp(logyhat3)
lines(a, yhat3, lwd=2,col='red')
L4 <- lm(log(C6H6) ~ log(CO))
plot(log(CO), log(C6H6), main="Log-Log", pch=16)
logyhat4 <- L4$coef[1] + L4$coef[2]*log(a)
lines(log(a), logyhat4, lwd=2,col='red')
CO=airq$`CO(GT)`
C6H6=airq$`C6H6(GT)`
n <- length(CO)
e1 <- e2 <- e3 <- e4 <- numeric(n)
# for n-fold cross validation
# fit models on leave-one-out samples
for (k in 1:n) {
y <- C6H6[-k]
x <- CO[-k]
J1 <- lm(y ~ x)
```

```
yhat1 <- J1$coef[1] + J1$coef[2] * C6H6[k] e1[k]
<- C6H6[k] - yhat1 J2 <- lm(y ~ x + I(x^2))
yhat2 <- J2$coef[1] + J2$coef[2] * C6H6[k] + J2$coef[3] * C6H6[k]^2
e2[k] <- C6H6[k] -yhat2
J3 <- lm(log(y) ~ x)
logyhat3 <- J3$coef[1]+ J3$coef[2] * C6H6[k]
yhat3 <- exp(logyhat3)
e3[k] <- C6H6[k] -yhat3
J4 <- lm(log(y) ~ log(x))
logyhat4 <- J4$coef[1] + J4$coef[2] * log(C6H6[k])
yhat4 <- exp(logyhat4)
e4[k] <- C6H6[k] - yhat4
}
cbind(e1=mean(e1^2), e2=mean(e2^2), e3=mean(e3^2), e4=mean(e4^2))
summary(L2)
confint(L2)
par(mfrow = c(2,2))
plot(L2)
```

PermutationTest

Meanvalueandobservationrangescheck

```
cbind(compound = c("NOx", "NO2","CO","NMHC","C6H6"), mean_value = c(mean(airq$
`NOx(GT)`), mean(airq$`NO2(GT)`), mean(airq$`CO(GT)`), mean(airq$`NMHC(GT)`),
mean(airq$`C6H6(GT)`)))
cbind(compound = c("S1-CO","S2-NMHC","S3-NOx","S4-NO2","S5-O3"), mean_value =
c(mean(airq$`PT08.S1(CO)`),mean(airq$`PT08.S2(NMHC)`),mean(airq$`PT08.S3(NOx)`),mean
(airq$`PT08.S4(NO2)`), mean(airq$`PT08.S5(O3)`)))

cbind(compound = c("NOx min", "NOx max", "NO2 min", "NO2 max"), range = c(
min(airq$`NOx(GT)`),max(airq$`NOx(GT)`), min(airq$`NO2(GT)`), max(airq$
`NO2(GT)`)))

cbind(compound = c("S2-NMHC min", "S2-NMHC max", "S3-NOx min", "S3-NOx max"), range = c
(min(airq$`PT08.S2(NMHC)`),max(airq$`PT08.S2(NMHC)`), min(airq$`PT08.S3(NOx)`), max
(airq$`PT08.S3(NOx)`)))
```

(1)S2-NMHC&S3-NOx

```
x0 <- sort(as.vector(airq$`PT08.S2(NMHC)`))
y0 <- sort(as.vector(airq$`PT08.S3(NOx)`))
n = 2*length(x0)
R <- 999
z0 <- c(x0, y0)
K <- 1:n
reps <- numeric(R)
t0 <- t.test(x0, y0)$statistic

D <- numeric(R)
options(warn = -1)
D0 <- ks.test(x0, y0, exact = FALSE)$statistic
```

```r
#number of replicates #pooled sample
#storage for replicates
for (i in 1:R) {
#generate indices k for the first sample
k <- sample(K, size = length(x0), replace = FALSE)
x0 <- z0[k]
y0 <- z0[-k] #complement of x1
reps[i] <- t.test(x0, y0)$statistic
D[i] <- ks.test(x0, y0, exact = FALSE)$statistic }
p0 <- mean(c(t0, reps) >= t0)
p <- mean(c(D0, D) >= D0)
options(warn = 0)
cbind(Test = c("T-test", "K-S test"), p.value = c(p0, p))
par(ps=11)
hist(reps, freq = FALSE, breaks = 20, xlab = "T-test statistics
(S2-NMHC&S3-NOx)",main = "Histogram of distribution of permutation T-test between
S2-NMHC&S3-NOx")
par(ps=8)
hist(D, freq = FALSE, breaks = 20, xlab = "K-S statistics (S2-NMHC&S3-NOx)",main =
"Histogram of distribution of permutation Kolmogorov-Smirnov test between S2-NMHC&S3-
NOx")
```

(2)NO2&NOx

```r
x2 <- sort(as.vector(airq$`NOx(GT)`))
y2 <- sort(as.vector(airq$`NO2(GT)`))
n = 2*length(x2)
R <- 999
z2 <- c(x2, y2)
K <- 1:n
reps1 <- numeric(R)
t2 <- t.test(x2, y2)$statistic
D1 <- numeric(R)
options(warn = -1)
D2 <- ks.test(x2, y2, exact = FALSE)$statistic #number
of replicates #pooled sample #storage for replicates
for (i in 1:R) {
#generate indices k for the first sample
k <- sample(K, size = length(x2), replace = FALSE)
x2 <- z2[k]
y2 <- z2[-k] #complement of x1
reps1[i] <- t.test(x2, y2)$statistic
D1[i] <- ks.test(x2, y2, exact = FALSE)$statistic
}
p1 <- mean(c(t2, reps1) >= t2)
p2 <- mean(c(D2, D1) >= D2)
options(warn = 0)
cbind(Test = c("T-test", "K-S test"), p.value = c(p1, p2))
```

```r
par(ps=12)
hist(reps1, freq = FALSE, breaks = 20, xlab = "T-test statistics (NO2&NOx)", main = "Histogram of distribution of permutation T-test between NOx&NO2")
par(ps=9)
hist(D1, freq = FALSE, breaks = 20, xlab = "K-S statistics (NO2&NOx)", main = "Histogram of distribution of permutation Kolmogorov-Smirnov test between NOx&NO2")
```

BayesianInference&MCIntegration

```r
O=airq$`PT08.S3(NOx)`; logO=log(O); sdO=sd(logO); me=mean(logO); hist(logO)
set.seed(1); m <- length(O); sigma=sdO; y <- rcauchy(m)
h<- (1/(sqrt(2*pi)*sigma))*exp(-0.5*((logO-y)/sigma)^2)
C <- mean(h)
post.mean <- mean(y*h)/mean(h)
C # estimate the normalizing constant
post.mean # estimate the posterior mean
me #the sample mean
num <- y*h # vector in the numerator
rc <- rep(0, times = m) # running c
rpm <- rep(0, times = m) # running posterior mean
for (i in seq.int(m)){ # seq.int(m) will be 1:m
 rc[i] <- mean(h[1:i])
 rpm[i] <- mean(num[1:i])/mean(h[1:i])
}
# now plot the results
par(mfrow = c(1,2))
plot(3:200, rpm[3:200], type = "l", ylim = c(0, 8))
lines(3:200, rc[3:200], type = "l")
plot(1:m, rpm, type = "l", ylim = c(0, 8))
lines(1:m, rc, type = "l")
set.seed(1) # make the experiment reproducible m
<- length(O) # number of simulated values k=500
t=rep(mean(logO),k)
post.mean=c=numeric(k)
sigma=seq(sdO*0.001,sdO*5,length.out = k)
for (i in 1:k){
y <- rcauchy(m)
h<- 1/sqrt(2*pi)*(1/sigma[i])*exp(-0.5*((logO-y)/sigma[i])^2) # compute h(theta)
C[i] <- mean(h) # estimate the normalizing constant post.mean[i] <- mean
(y*h)/mean(h) # estimate the posterior mean
}
plot(post.mean,type = "l")
lines(1:k,t,type = "l")
hist(post.mean,prob="TRUE")
```

MCMC

(1) Metropolis-Hastingssampler

```r
set.seed(000)
f <- function(x, O, sigma) {
if (any(O < 0) ) return (0)
stopifnot(sigma > 0)
```

```
return(1/sqrt(2*pi) * 1/sigma * exp(-0.5*((O-x)/sigma^2)))
}
k = 0
m = 1000
u = runif(m)
b = 201
index = 500:1000
a = ppoints(100)
QN = qnorm(a)

#NOx
sigma.nox=sd(airq$`NOx(GT)`)
nox = mean(airq$`NOx(GT)`)
x.nox = numeric(m)
x.nox[1] = rnorm(m,nox,sigma.nox)
options(warn = -1)
for (i in 2:m) {
xt.nox <- x.nox[i-1]
y.nox <- rnorm(m,nox,sigma.nox)
num.nox <- f(y.nox, nox, sigma.nox) * dnorm(xt.nox,nox,sigma.nox)
den.nox <- f(xt.nox, nox, sigma.nox) * dnorm(y.nox,nox,sigma.nox)
if (u[i] <= num.nox/den.nox) x.nox[i] <- y.nox else {
 x.nox[i] <- xt.nox
 k <- k+1
 }
}
options(warn = 0)

cbind(Accept.rate = 1-k/m)
Y1.nox = x.nox[index]
plot(index, Y1.nox, type = "l", main = "Markov Chain for NOx", ylab = "NOx")


Y.nox = x.nox[b:m]
Q.nox = quantile(x.nox,a)
qqplot(QN, Q.nox, main = "QQ plot of NOx", xlab = "Normal Quantiles", ylab = "Sample Quantiles")
qqline(x.nox, distribution = qnorm, qtype = 7, col="blue", lty = 6)
hist(Y.nox, breaks=30, xlab="", freq=FALSE, main = "Histogram of NOx Sample
distribution")
```

(2) RandomWalkMetropolis

```
set.seed(000)
library(knitr)
library(rmutil)
rw.Metropolis <- function(n, sigma, x0, N) {
 x <- numeric(N)
 x[1] <- x0
 u <- runif(N)
 k<-0
```

```
for (i in 2:N) {
y <- rnorm(1, x[i-1], sigma)
if (u[i] <= (dlaplace(y, n) / dlaplace(x[i-1], n)))
x[i] <- y
else {
x[i] <- x[i-1]
k<-k+1
}
}
return(list(x=x, k=k))
}

n <- 4 #degrees of freedom for target Student t dist.
N <- 2000
sigma.no2 =sd(airq$`NO2(GT)`)
sigma.nox = sd(airq$`NOx(GT)`)
sigma.co = sd(airq$`CO(GT)`)
rw.no2 <- rw.Metropolis(n, sigma.no2, mean(airq$`NO2(GT)`), N)
rw.nox <- rw.Metropolis(n, sigma.nox, mean(airq$`NOx(GT)`), N)
rw.co <- rw.Metropolis(n, sigma.co, mean(airq$`CO(GT)`), N)
# acceptance rate for each chain

a <- c(.05, seq(.1, .9, .1), .95)
Q <- qlaplace(a)
mc.no2 <- rw.no2$x[501:N]
mc.nox <- rw.nox$x[501:N]
mc.co <- rw.co$x[501:N]
plot(501:N, mc.co, type = "l", main = "Markov Chain for CO", ylab = "CO")
plot(501:N, mc.no2, type = "l", main = "Markov Chain for NO2", ylab = "NO2")
plot(501:N, mc.nox, type = "l", main = "Markov Chain for NOx", ylab = "NOx")
Qrw.NO2 =quantile(mc.no2,a)
Qrw.NOx =quantile(mc.nox,a)
Qrw.CO =quantile(mc.co,a)
result = round(cbind(Q, Qrw.NO2, Qrw.NOx, Qrw.CO), 4)
name = c( "Q","NO2", "NOx", "CO")
accept.rate = round(cbind(1 - rw.no2$k /N, 1-rw.nox$k/N, 1-rw.co$k/N),4)
accept.name = c("Accept rate for NO2", "Accept rate for NOx", "Accept rate for CO")
kable(result, col.names = name)
kable(accept.rate, col.names = accept.name)
variance = cbind(var(airq$`NO2(GT)`), var(airq$`NOx(GT)`), var(airq$`CO(GT)`))
var.name = c("Variance for NO2", "Variance for NOx", "Variance for CO")
kable(variance, col.names = var.name)
```

II.Limitations:

a. Ourprojectonlyanalyzedtheone-yearairqualitydataset.However,themethodsweutilizedin theprojectcanbe furtherappliedinanalyzingotherexperimentaldataandobtainingthe statisticaldistributionforrawdata.

b. Becauseourrawdatacontainsmanymissingvalues,aftercleaningprocess,manyobservations in differenttimepointswereeliminated.Duetothediscretetimeline,analysisislimitedto

certainmarkovchain.

c. Someofthevariables(C6H6,NOx(GT),RH,andPT08.S1)areneitherlognormalornormal distribution.However,sincewecannotdealwiththosedistributions,weanalyzeitwith normal distribution.

III.Futureapplications:

a. Wearelookingforwardtoapplyingourmethodsinlong-termdatarecordsinlargertimescale. Ifacquiringrecords eachadditionalyear,wecanperformanalysisonyearlyairpollutants distributionandcorrelationsamongcertain variablesandcompareannualresults.

b. Basedoncomparisonsamongannualairqualityanalysis,wemaydrawconclusionsonthe trendofairquality improvement/deteriorationandgiveadvicetoenvironmentalpollution management.

c. Wecancollectdatafromindustrialmetalsmeltinginthecityandcorrelatewithairquality datatofindout therelationshipbetweenairpollutiontocertainmetalproductionprocess.We cantargetcertain industrialmetaloxidation/reductionprocessesthatcontributethemostto airpollutantsandgiveadvice aimingatoptimizingindustrialprocessingmethods.