7-18-2019

Autocorrelation Reduction and Consistency Analysis of Correlation for Human Brain Activity Data

Xiaowen Zhou
xz6955@rit.edu

R·I·T

Autocorrelation Reduction and Consistency Analysis of Correlation for Human Brain Activity Data

by

Xiaowen Zhou

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Applied Statistics
School of Mathematical Sciences, College of Science

Rochester Institute of Technology
Rochester, NY
July 18, 2019

Committee Approval:

Peter Bajorski Date

Professor, School of Mathematical Sciences
Thesis Advisor

Linlin Chen Date

Associate Professor, School of Mathematical Sciences
Committee Member

Minh Pham Date

Assistant Professor, School of Mathematical Sciences
Committee Member

ABSTRACT

In recent years, the number of studies using functional magnetic resonance imaging (fMRI) on human brain activity has increased rapidly, which has become a hot topic in medical and academic fields. The autocorrelation and correlation problems in the time series of human brain activity have also become an important research direction. It is found that there are relative residuals in the time series of human brain activity processed by smoothing splines. To solve this problem, B-spline is used to smooth the curve. By choosing the right knots, a better smoothness method to process the human brain activity data is provided. In addition, the study also found that the time series of human brain activity has correlations. The multiple scans of the same person were analyzed to see if these correlations were consistent. In order to evaluate this point, correlation is used as a response variable Y and person as a factor X to fit a random effect model. By calculating the percentage of variation in Y to determine whether the scans are similar to each other. The results show that the mean-centering time series data with 0^{th} order difference has the most consistent correlation.

ACKNOWLEDGMENTS

I would first like to express my sincere gratitude to my thesis advisor, Dr. Peter Bajorski, for his greatly support and help in this research. This thesis would not have been materialized without his guidance and encouragement.

I would like to thank Dr. Linlin Chen and Dr. Minh Pham for their kind advice, as well as being my thesis committees.

I would also like to thank Dr. Robert Parody for his kind advice and help.

I would also like to thank Shawna Hayes for the care and assistance, and all my friends for their help during my graduate study.

Last but not least, I want to thank my parents and family for their endless love and support.

Data were provided [in part] by the Human Connectome Project, WU-Minn Consortium (Principal Investigators: David Van Essen and Kamil Ugurbil; 1U54MH091657) funded by the 16 NIH Institutes and Centers that support the NIH Blueprint for Neuroscience Research; and by the McDonnell Center for Systems Neuroscience at Washington University.

CONTENTS

List of Figures

List of Tables

1 Introduction

Functional magnetic resonance imaging (fMRI) is one of the best tools for the human to study brain activity. Scientists use neuroimaging processing to extract frequency data of different brain regions for further analysis and intensive study. The research of fMRI data has produced a large number of noise data with a complex correlation structure. An important role was played by statistics in analyzing the nature of the large size, high-dimensional and noisy data and interpreting the results. The large data size, high dimensionality, and noise of fMRI data make statistical analysis challenging (Chen, 2015).

The linear parameter regression model of fMRI time series data has autocorrelated residuals (Carew et al., 2002). It will violate the independence hypothesis if the residuals have an autocorrelation structure; also, the estimated standard error will be biased. This may lead to the incorrect degree of freedom and the expansion of test statistics. Therefore, scientists conducting fMRI studies must deal with autocorrelations in order to make effective statistical inferences. Two general methods for dealing with these autocorrelations are smoothing and whitening (Carew et al., 2002). It is very difficult to obtain an accurate estimation of intrinsic autocorrelation. Friston et al. (2000) state that the better way to minimize bias was to smooth rather than whiten the data. Therefore, it is prudent to study the applicability of various smoothing methods for fMRI data.

In statistics and image processing, smoothing a data set is mainly to grasp the main patterns in the data by establishing approximation functions and remove noise, structural details and instantaneous phenomena (Wikipedia, 2019). In the smoothing process, the signal data points are modified. The individual data points generated by noise are reduced, and the points lower than

the adjacent data points are raised to obtain a smoother signal. Smoothing can be used in data analysis in two important forms. First, if the assumptions of smoothing are reasonable, more information can be obtained from the data. Second, providing a flexible and robust analysis. There are many different algorithms for smoothing. This paper focuses on the use of B-spline as the main smoothing tool in time series analysis of human brain activity. By choosing the right knots, finding the most suitable smoothness for the time series of human brain activity data.

2 Check the Autocorrelation

2.1 Data Structure

The initial experiment included 820 people scanned by functional magnetic resonance imaging (fMRI) and four times for each person. Through the complex imaging processing, the fMRI images were converted into frequency data. In this process, the original fMRI images were identified as 116 brain regions, and each brain region contains 1200 frequency data points. The full data is stored in a four-dimension array with the following dimensions:

- [1200 (points in time), 116 (brain regions), 4 (scans) and 820 (people)].

At present, a small subset of all data was used to concentrate on specifics. There are 4 scans for each person, and 3 people for this small subset. Load both the original data and smoothed data for analysis. They both are stored in four-dimension arrays [1200, 116, 4, 3].

2.2 Residual Analysis

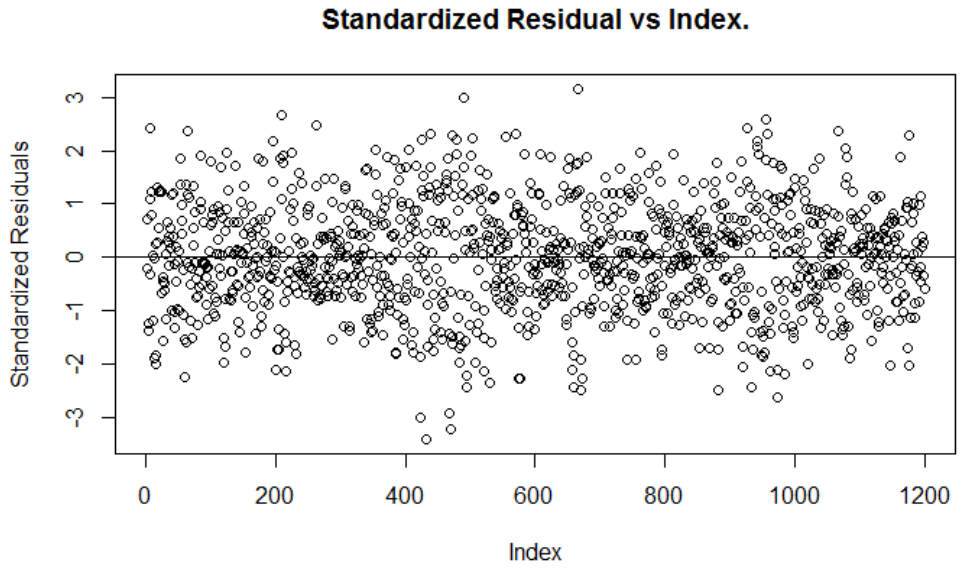


Figure 1. Residual plot (Subset.Resid.arr[,1,1,1])

Based on the residual plot above, the residuals are randomly dispersed around the horizontal axis. And it shows no systematic nor other special patterns. That means the residuals are pretty random and independent.

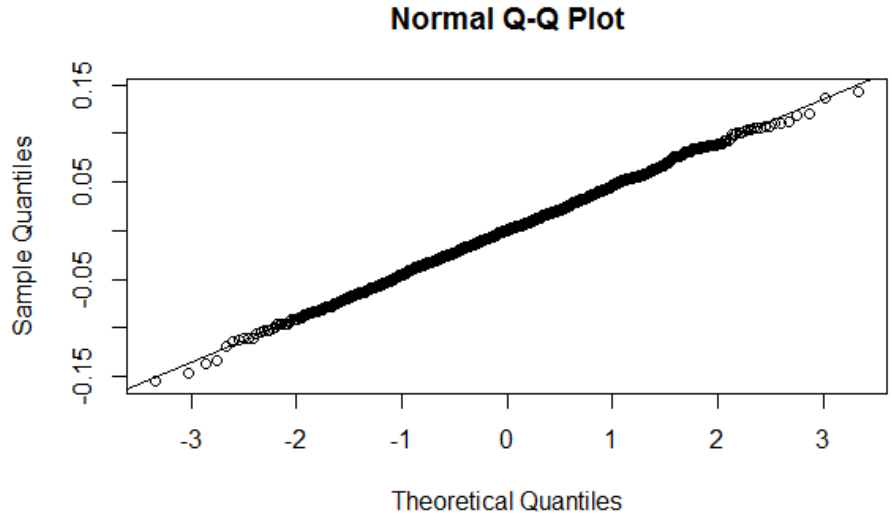


Figure 2. Normal Probability Plot of Residual

From the normal probability plot of the residuals, it shows that the residuals are close to the diagonal line which represents the normal distribution. It is reasonable to assume that the residuals are normally distributed.

To check the variability for the residuals, the control chart could be used. The X-Bar chart shows how much variation exists in the process over time. This chart is used to monitor the mean of a process based on samples collected from the process at a given time (such as hours, days, months, years, etc.). The measurement of the sample at a given time constitutes a subgroup. Usually, an initial series of subgroups is used to estimate the mean and standard deviation of a process. Then, the mean and standard deviation are used to generate control limits for the average value of each subgroup (NCSS, 2019). Control limits reflect the actual amount of variation that is observed. Assuming that the variation can be described by the normal distribution. It means that 99.73% of all of our observations will fall somewhere between three standard deviations below the mean (-3σ) and three standard deviations above the mean ($+3\sigma$). Here, this principle was used to set as control limits.

The ‘qcc’ function in R library could be used to generate the X-Bar control chart.

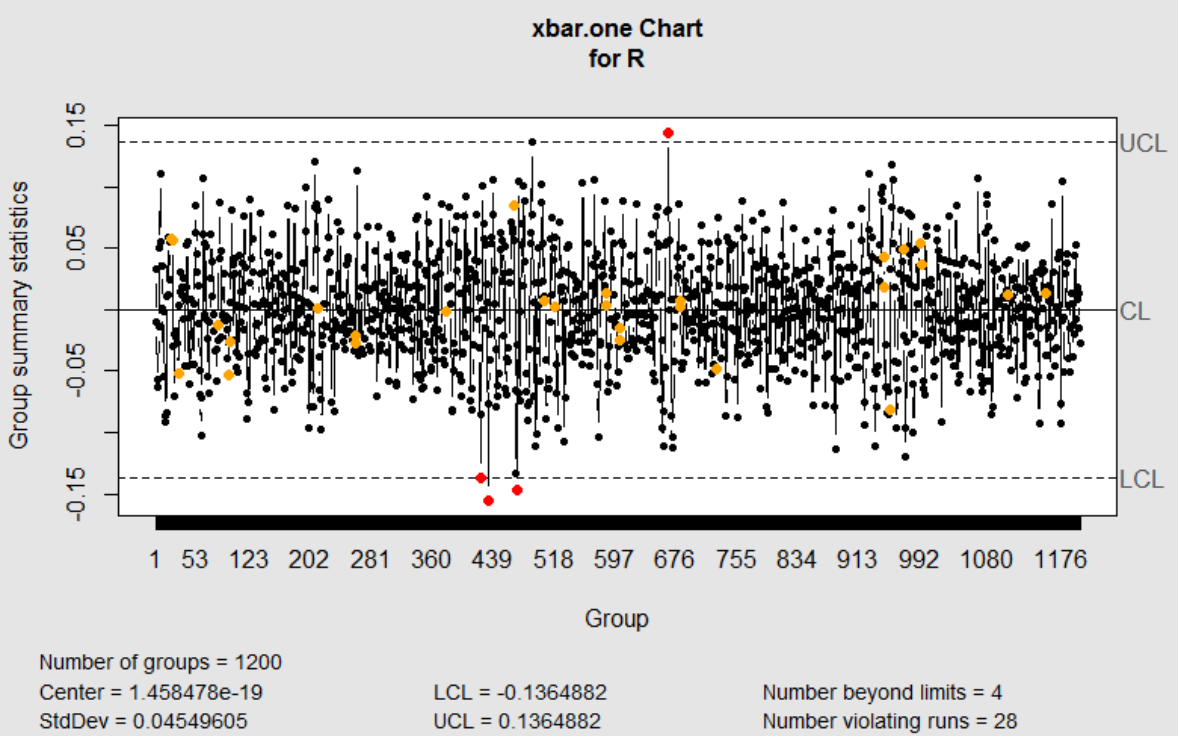


Figure 3. X-Bar Control Chart

From Figure 3, the out-of-control points were highlighted in red. The standard deviation was reported in the legend for the chart. It is easy to see that one measurement around 676 was above the Upper Control Limit (UCL) and three measurements between 439 and 518 were below the Lower Control Limit (LCL). In general, it is in control, because there were only a few 4 out of over 1000 observations.

2.3 Test of Randomness

Runs test can be used to determine whether data sets come from random processes (Bradley, 1968). A run is defined as a series of increasing or decreasing values. The number of increasing or decreasing values is the length of the run. In random data sets, it follows a binomial distribution when the probability of the $(i+1)_{th}$ value is greater or less than the i_{th} value. This forms the basis of runs test (NIST, 2012).

Runs test is a test for the randomness of a numeric sequence X by studying the runs frequency R. Usually, by comparing each element of a sequence with its median, each numeric sequence can be converted into binary data defined as 1 and 0. Given n 1 and m 0, the runs R is defined as a series of similar responses with a statistical distribution. On the basis of known distribution, the exact p-value of small sample data can be calculated. When the sample size is large, one can use the normal approximation with mean $\mu = \frac{2mn}{m+n} + 1$ and variance $\sigma = \frac{2mn(2mn-m-n)}{(m+n)^2*(m+n-1)}$.

Here we are not sure how the process will deviate from randomness, then the possibility of either a cyclic effect or a trend effect should be allowed. In this case, if $r \geq c$ *or* $r \leq c$**,** the null hypothesis of randomness should be rejected, and the alternative hypothesis of either a cyclic effect or a trend effect should not be rejected (PSU, 2018)**.**

```
Runs Test

data:  R
statistic = -8.2596, runs = 458, n1 = 600, n2 = 600, n = 1200, p-value < 2.2e-16
alternative hypothesis: nonrandomness
```

Table 1. Runs Test

Based on the table 2.3.1, the p-value is very small, which means nonrandom. So, the nature of this non-randomness need to be explored. Trying to plot the residuals vs. fitted values (smoothed values), and check if e_i depends on e_{i-1}.

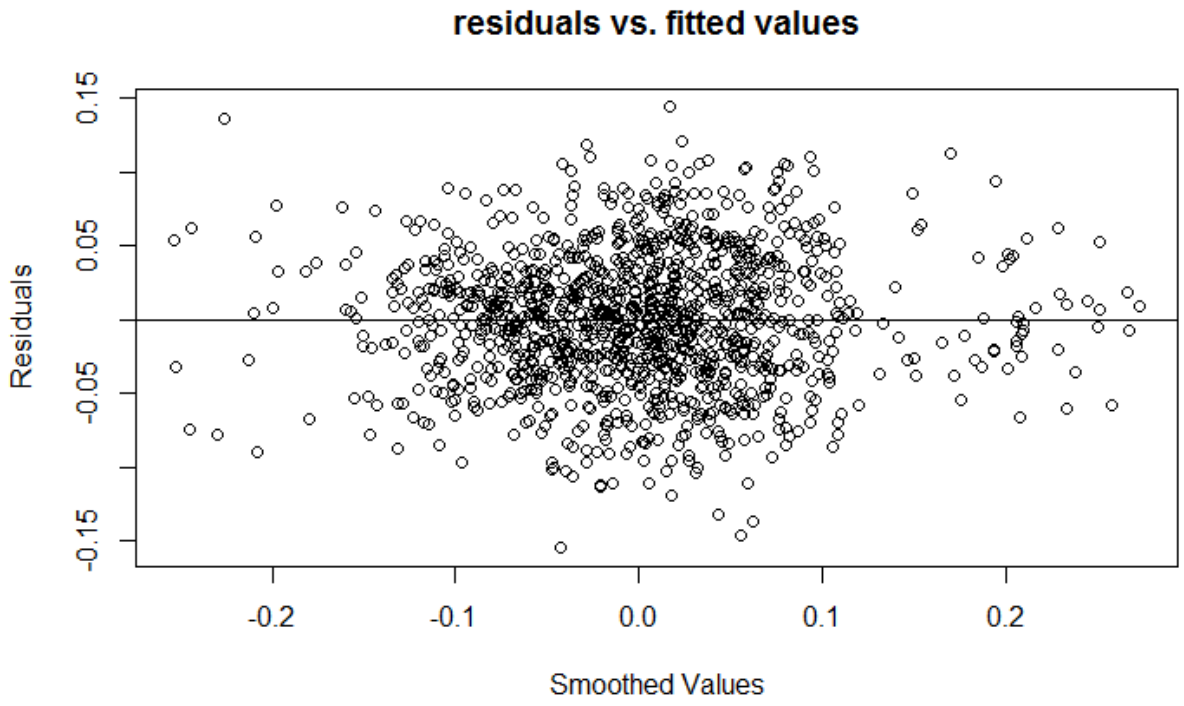


Figure 4. Residuals vs. Fitted values

From the plot above, the residuals seem pretty symmetrically distributed, tending to cluster towards the middle of the plot. It looks a little spreading when the fitted value increases around 0. And notice that as the smoothed value continuous increases, there is a narrow down and centralized trend of the residuals around the zero residuals line. This is to be expected because there are fewer observations at the end. It indicates that the non-constant variances problem is not very obvious. So, the patterns could identify the error is random.

2.4 Autocorrelation Check

2.4.1 The Autocorrelation Function of Residuals

When the data is a time series, the autocorrelation function (ACF) is widely used to do the analysis. Specifically, the ACF shows the correlation between the points separated by different time lags. The representation is ACF (N, the number of time periods between points) = the

correlation between the points separated by N time periods. For example, ACF(0)=1 means all data are completely correlated with themselves; ACF(1)=0.7 means the correlation between one point and the next point is 0.7. Therefore, ACF indicates how relevant points are with each other, based on the number of time steps they are separated by. This is the point of autocorrelation, which is the correlation between past data points and future data points, for different values of time intervals. Normally, the ACF is expected to drop to zero as the points become more separated (N becomes larger) because it is usually difficult to predict the future by the future from a given set of data.

The ACF will reveal if there is any autocorrelation in the residuals. The following plot shows an ACF of the residuals from the brain activity time series data.

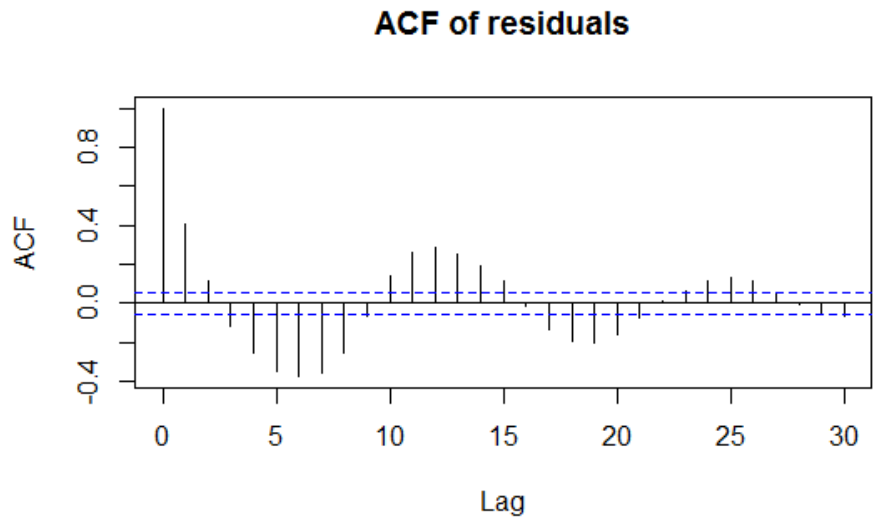


Figure 5. ACF of Residuals

There is a lot of significant spikes are seen in the ACF plot above. It suggests that a large amount of autocorrelation in the residuals.

2.4.2 Durbin-Watson Test

Another test of autocorrelation that is designed to take account of the regression model is the Durbin-Watson (DW) test. It is used to test the hypothesis that there is no lag one autocorrelation in the residuals. If there is no autocorrelation, the DW distribution is symmetric around 2. Most computer packages will report the DW statistic automatically, and should also provide a p-value. A small p-value indicates that there is a significant autocorrelation remaining in the residuals. For the brain activity data, based on the table below, the DW value is less than 2 which means there is some positive autocorrelation in the residuals. And the p-value is really small. So, the DW test reveals some significant lag one autocorrelations.

```
Durbin-Watson test

data:  R ~ Index
DW = 1.1941, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is not 0
```

Table 2. Durbin-Watson test

Both the ACF plot and the DW test show that some autocorrelations remaining in the residuals. This means some information remaining in the residuals that can be exploited to obtain better forecasts. So, other ways of smoothing need to be explored, so that the autocorrelation effect could be minimized while keeping other good properties.

3 Smoothing the Data

3.1 B-spline

B-spline constitutes an attractive method for nonparametric estimation of a series of statistical objects of interest (Racine, 2018). A spline function is a function constructed piecewise from polynomial functions. This term comes from a tool that drafters use to create smooth shapes with the desired characteristics. Racine (2018) argue that drafters have long used a flexible strip that

fixed in a location at multiple points. By relaxing these points, a smooth curve passing through them was formed. Combining the malleability of spline material with the constraints of control points will minimize the energy required to bend the strip between the fixed points, which will be the smoothest and most reasonable shape. This class of splines is "B splines", which is short for “basis spline".

The B-spline function is a combination of flexible strips passed through the amount of control points and constructs smooth curves. These functions allow multiple points to be used to create and manage complex shapes and surfaces. Therefore, B-spline function and Bezier function are widely used in shape optimization methods (Talebitooti, 2015). The B-spline is defined by its order (n) and the number of internal knots (k). Because there are two endpoints as themselves knots, the total number of knots will be k+2. The degree of B-spline polynomial would be spline order n-1, that is, degree = n-1 (Racine, 2018).

B-spline is a generalization of Bezier curve, that is, a B-spline without internal knots is a Bezier curve. So, the Bezier curve can be expressed as well as the B-spline. Moreover, because B-spline introduces the feature that the number of control points is independent of the curve order, the control ability for the lower order curve could be better. Therefore, B-spline has a larger degree of freedom and can define many control points without worrying about the high order of curves causing the difficulty of calculation. Compared with Bezier curve, B-spline curve can control the shape of the curve better. If the same control point is defined, the lower curve order is, the farther the curve is from the point. If the spline curve needs to be transformed, only the sequence of control points of B-spline needs to be transformed. The parameter interval and the number of times remain unchanged. These are very good qualities in practical applications. Users usually get more control points by adding knots and increasing the number of knots. Furthermore,

without affecting the overall trend of the curve, the partial modification mode of B-spline is used to specify and adjust the curve.

The B-spline function has been widely used in many fields such as computer-aided design, imaging processing, manufacturing, numerical control, etc. The most challenging task in these applications is to determine the number of knots and their corresponding locations (Dung and Tjahjowidodo, 2017). In this paper, the first difficulty is to select different sets of knots for locations and continuity levels. Secondly, choose the right locations of knots, analyze the residuals and autocorrelation. The ultimate goal is to minimize autocorrelation.

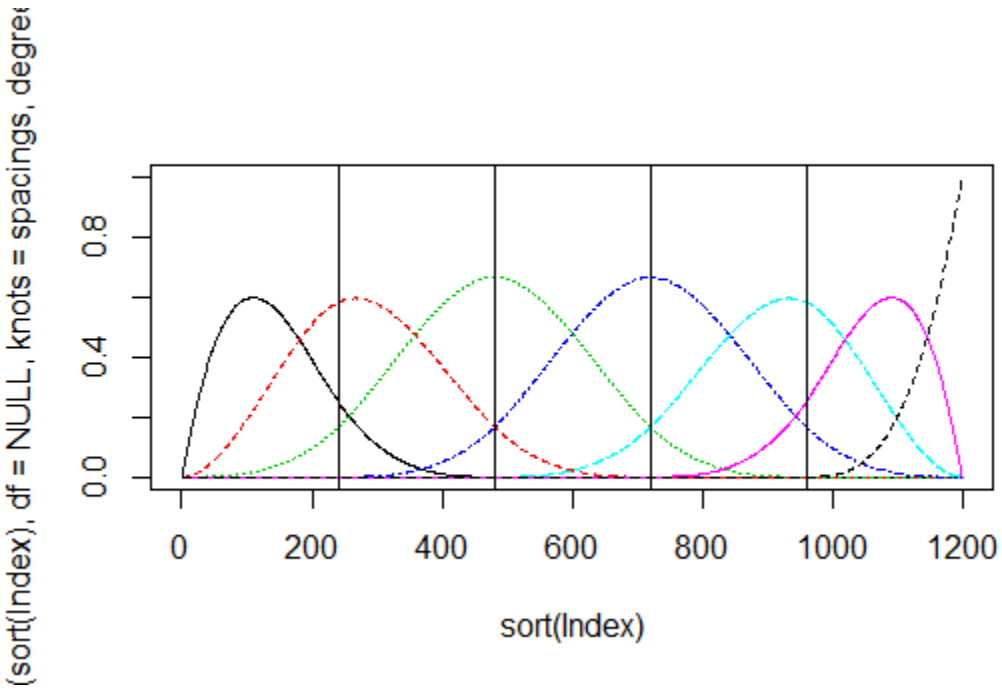


Figure 6. B-spline with 4 knots and equidistance

The figure above is a B-spline plot with taking 4 knots of equal distance.

3.2 Select and Reduce the Number of Knots

3.2.1 Find a Large Number of Knots

Identify knots at all the maximum and minimum points through all observations.

Specifying knots at the maximum and minimum locations as follows:

Maximum location: ss[i-1]<ss[i] & ss[i+1]<ss[i]

Minimum location: ss[i-1]>ss[i] & ss[i+1]>ss[i]

The number of observations is 1200, and the range of i is from 2 to 1200. The total number of knots at the maximum and minimum location are 650, which is shown in following R code.

```
Index <- 1:1200
ss<-Subset.Scans.arr[,1,1,1]
i<-2:1200
location<-as.numeric(which((ss[i-1]<ss[i] & ss[i+1]<ss[i])|(ss[i-1]>ss[i] &
      ss[i+1]>ss[i])))
length(location)
[1] 650
```

Use B-spline function in R and run bs() smoothing. The bs() function in R produces B-splines, a computationally efficient way to compute cubic regression splines. The set of knots that was calculated before can be used in the bs() function.

```
require(splines)
regr.spline <- lm(ss ~ bs(Index, df = NULL, knots=location, degree = 3,
intercept=T))
```

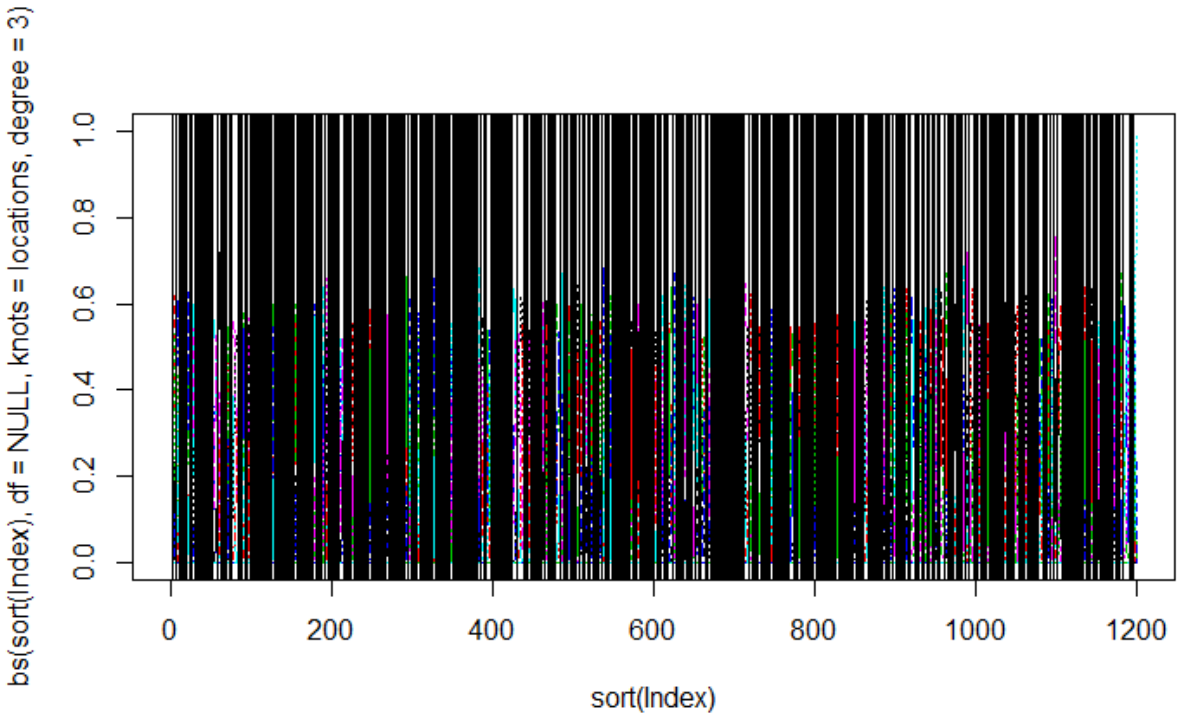


Figure 7. B-spline with 650 knots

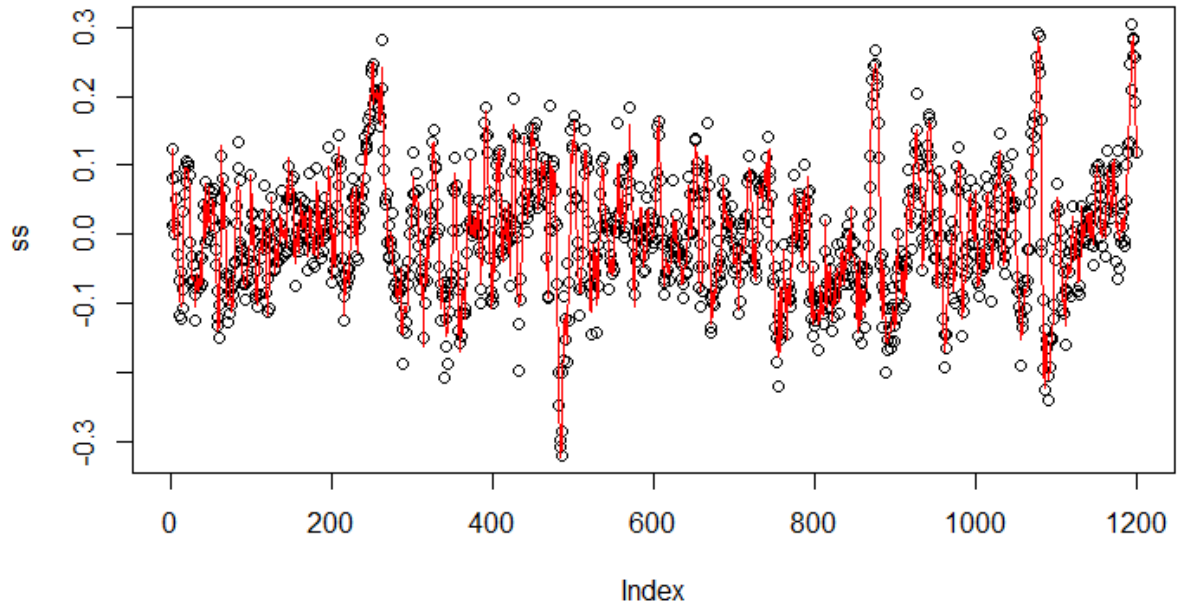


Figure 8. Fitting model plot

The figure above is fitting models with splines by adding the fitted curve based on the model to a scatter plot. This is likely to be overfitting the data. Therefore, the number of knots should be reduced appropriately.

3.2.2 Reducing the Number of Knots

Identify the local minimum and maximum from the smoothed data with an additional rule that they should not be too close to each other. Assume the knots should be at least 5 intervals apart. By using a loop, check for at least 5 intervals apart (knots locations going from left to right), and eliminating the knot. The selection R code is shown below:

```
count=0
n=5       # set n is 5 intervals apart
for (j in 1:(nrow(set.knots)-1)){
   j=j-count
   #the knots should be at least 5 intervals apart
   #the minima or maxima location minus the previous one is greater than 5
   l = set.knots$location[j+1]-set.knots$location[j]
   if (l<n){
   #delete the knot which is too close to the previous one
     set.knots=set.knots[-(j+1),]
     count=count+1
   }
   else{
     set.knots=set.knots
   }
 }
set.knots
```

	location	y.value
1	3	0.014426723
2	6	0.050761807
3	10	-0.067636453
4	11	-0.118271491
5	12	-0.053882877
6	16	0.033326083
7	17	0.096250022
8	19	0.076560447
9	24	-0.012663277
10	25	-0.067493973
11	29	-0.086621442
12	31	-0.078014426
13	32	-0.057827205

	location	y.value
1	3	0.014426723
3	10	-0.067636453
6	16	0.033326083
9	24	-0.012663277
11	29	-0.086621442
14	34	-0.069494623
18	39	-0.071543165
23	44	0.038635854
27	49	0.044520923
29	59	-0.131472162
32	64	0.015300491
37	69	-0.056164945
40	74	-0.114796412

Figure 9. Select new set of knots (5 intervals apart)

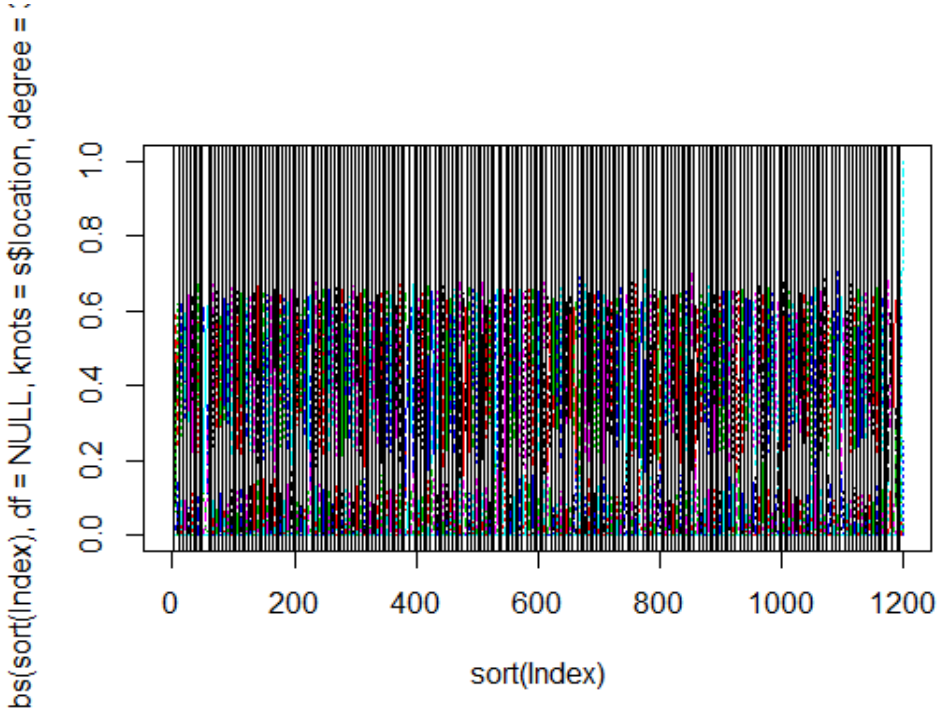


Figure 10. B-spline with reduced knots

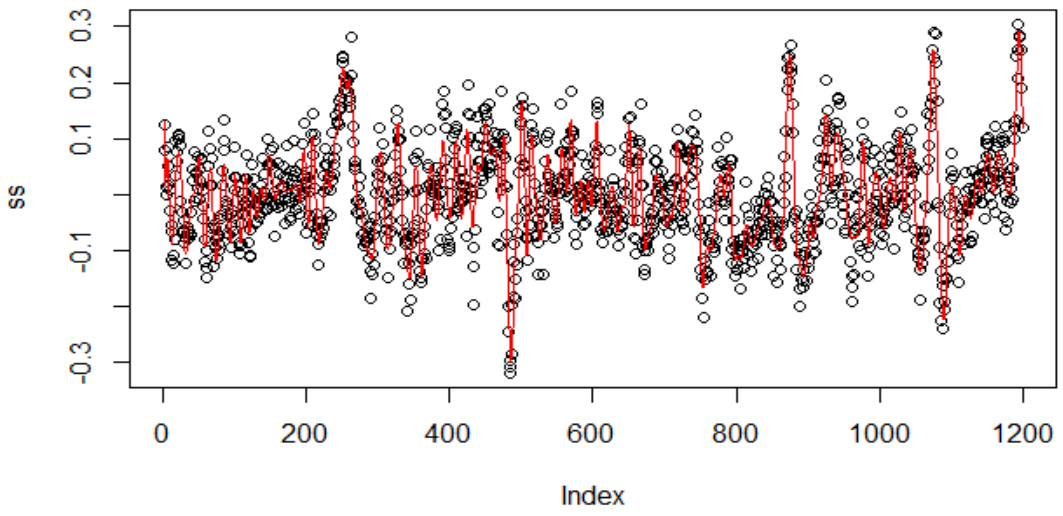


Figure 11. Fitting model plot

Based on the plots above, the number of knots was reduced a little bit.

In order to test whether the B-spline smoothing method and the selected knots provide a better smoothing, residual analysis and autocorrelation check are necessary.

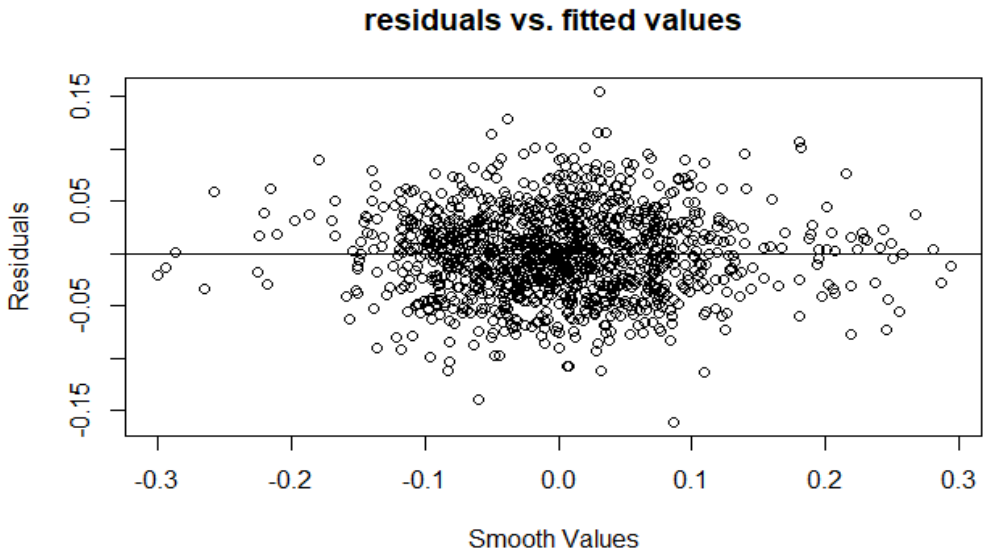


Figure 12. Residuals vs. Fitted value plot (B-spline method and reduced knots)

From the residuals vs. fitted values plot, the residuals seem pretty symmetrically distributed, tending to cluster towards the middle of the plot. It looks random.

```
        Durbin-Watson test (B-spline smoothing and reduced knots)

data:  res ~ Index
DW = 1.471, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is not 0
```

Table 3. Durbin-Watson test (B-spline smoothing and reduced knots)

```
        Durbin-Watson test (original smoothing)
data:  R ~ Index
DW = 1.1941, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is not 0
```

Table 4. Durbin-Watson test (original smoothing)

If there is no autocorrelation, the DW distribution is symmetric around 2. Based on the two tables above, the DW value both are less than 2 which means there are some positive autocorrelations in the residuals. But with the knots reduced, the DW value became a little larger than the original data. And the p-value is still really small. So, the DW test still reveals some significant lag one autocorrelation.

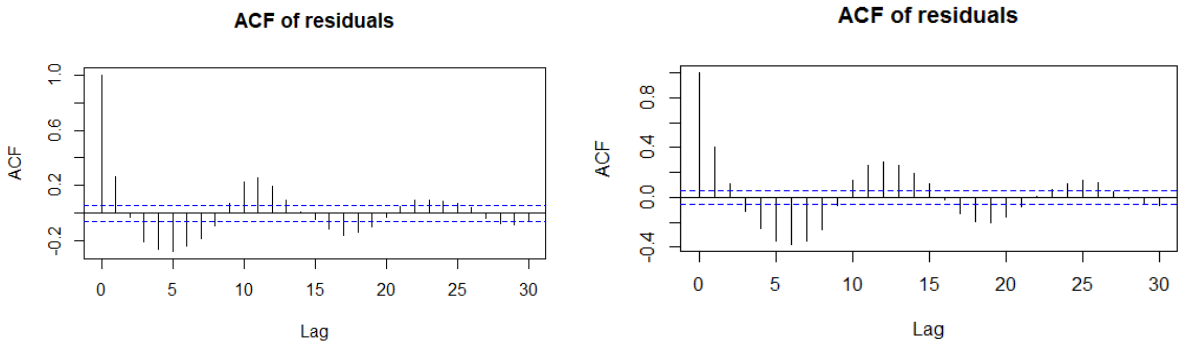


Figure 13. ACF comparison plot (L: new smoothed; R: original smoothed)

Compared to the plot of original smoothed data (the plot on the right side), the value of significant spikes in the ACF plots are decreased. This might suggest that the number of autocorrelations in the residuals are reduced.

3.2.3 Further Reduction of Knots

First of all, calculate the baseline of the correlation value. Then, use a loop in R code to conduct the reduction. For each knot, check how much the correlation between consecutive residuals is reduced when the knot was removed.

	k.loc	correlation	reduction
1	3	0.2694287	-0.10109171
2	10	0.2732231	-0.10488613
3	16	0.2668010	-0.09846399
4	24	0.2659599	-0.09762288
5	29	0.2660078	-0.09767082
6	34	0.2645059	-0.09616890
7	39	0.2648070	-0.09647002
8	44	0.2660725	-0.09773555
9	49	0.2701158	-0.10177884
10	59	0.2776577	-0.10932072

Table 5. Knots reduction information

Remove the knot which gives the best (highest) reduction in correlations.

```
# search for the largest positive reduction
subset(correlation.a, reduction > 0)
[1] k.loc       correlation reduction
    <0 rows> (or 0-length row.names)
# choose the highest reduction in correlations
cor.max<-which.max(correlation.a$correlation)
cor.max
[1] 144
value<-correlation.a[cor.max,]
value
    k.loc correlation  reduction
144   826   0.2644662 -0.0961923
```

Based on the R code above, there is no positive reduction after selecting knots which should be at least 5 intervals apart. So the rule of selection might be changed a little. In order to increase the number of knots selected, we reduced n to 4, which means the knots would be at least 4 intervals apart. Therefore, change the n=4, and run the selection R code again to choose a new set of knots. The number of knots increased from 206 to 246. After running the DW test, the DW value increased from 1.471 to 1.6634, which became closer to 2. Besides, the correlation value reduced a little, from 0.2644661 to 0.1682831, compared to those when n=5. Now we need to continue checking the correlation and reduction between consecutive residuals.

```
subset(correlation.a, reduction > 0)
    k.loc correlation    reduction
92    442   0.1682804 2.769645e-06
99    472   0.1682415 4.158100e-05
140   675   0.1682821 9.839158e-07
155   748   0.1682802 2.945366e-06
174   834   0.1682811 2.065723e-06
175   838   0.1682819 1.264286e-06
198   961   0.1682731 1.002078e-05
199   966   0.1682601 2.303033e-05
```

Table 6. A subset of the positive reduction

Based on the table above, some positive reductions can be seen when selecting knots at least 4 intervals apart. So, the largest reduction 4.1581e-05 could be selected and this knot was deleted. After that, check the correlation and DW value after removing the knot which gives the max reduction. The DW value increased from 1.471 to 1.6635 which is larger than that before, and more close to 2. The correlation value decreased a little, from 0.1682831 to 0.1682415.

Back to previous steps until no further reduction can be achieved. Based on the result table below, after reducing the number of knots and checking the correlation, the value keeps decreasing until removed 6 knots. After that, the value became increase again. Also, the DW

value became a little larger (from 1.6634 to 1.6636) when 6 knots were removed. It shows that the autocorrelation in the residuals is reduced after deleting 6 knots.

N of knots removed	1	2	3	4	5	6	7	8
DW	1.6635	1.6635	1.6635	1.6635	1.6635	1.6636	1.6635	1.6635
cor	0.1682415	0.1682185	0.1682156	0.1682132	0.1682111	0.1682101	0.1682104	0.1682109

Table 7. Autocorrelation comparison (4 intervals apart)

From Table 7, there is no further reduction that can be achieved after 6 knots have been deleted, and this is based on the rule of knots selection that knots were at least 4 intervals apart. However, the correlation value was reduced to 0.1682101 which was still not close to 0. In addition, the DW value was increased to 1.6636, but it still less than 2. So, more knots might be added, and see what happens.

In order to increase the number of knots, n will be reduced to 3, which means the knots would be at least 3 intervals apart. The number of knots increased from 246 to 308.

```
Durbin-Watson test

data:  res ~ Index
DW = 2.1261, p-value = 0.03106
alternative hypothesis: true autocorrelation is not 0
```

Table 8. Durbin-Watson test (3 intervals apart)

Table 8 shows the DW value was increased to 2.1261, which was close to 2. The correlation value reduced a lot, from 0.1682101 to -0.06303416, which was almost 0. Then, the correlation and reduction between consecutive residuals still need to be checked.

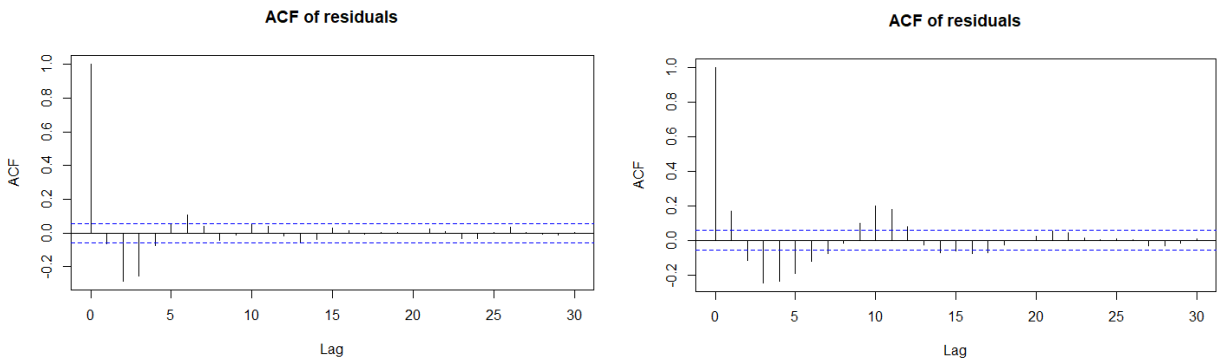


Figure 14. ACF comparison plot (L: n=3; R: n=4)

Compared to the right ACF plot (when n=4), the value of significant spikes decreased in the left ACF plot (n=3). This states that the amount of autocorrelation in the residuals was reduced. Furthermore, by comparing to the plot of original smoothed data (Figure 13, the plot on the right side), the autocorrelations in the residuals are reduced a lot. It might indicate that the B-spline smoothing method with choosing the right set of knots works well.

N of knots removed	1	2	3	4	5	6	7	8
DW	2.1264	2.1266	2.1266	2.1267	2.1267	2.1267	2.1267	2.1268
cor	-0.06320119	-0.06327865	-0.0633056	-0.06333018	-0.0633496	-0.06336668	-0.0633 7282	-0.0633 7849

Table 9. Autocorrelation comparison (3 intervals apart)

After reducing the number of knots when n=3, the absolute value of correlation keeps increasing (became far from 0). Besides, the DW value still getting larger (greater than 2) when the knots were removed. It shows that the autocorrelation in the residuals is increased after deleting knots.

Try to keep increasing the number of knots and reduced n to 2, which means the knots would be at least 2 intervals apart. The number of knots increased from 308 to 419. After doing the DW

test and calculation the correlation, the DW value (2.521) and the absolute value of correlation (-0.2605046) were both increased. Based on the results, the number of knots should be kept when choosing the knots at least 3 intervals apart (n=3), which could give the best DW value and the smallest correlation value.

4 Correlation Analysis

4.1 Centered Data

In regression problems and some machine learning algorithms, as well as in the process of training neural networks, the original data usually needs to be centralized (Zero-centered or Mean-subtraction) and standardized (Standardization or Normalization). Centering is to subtract the mean from the original data, while standardization is to subtract the mean from the original data and divide it by standard deviation. The data obtained contains the mean is 0 and the standard deviation is 1. Data centralization aims to unify the scales of data with different variables. This is a basic work of data mining. Because different evaluation indicators often have different dimensions and units, this situation frequently affects the results of data analysis. To eliminate the dimension effect between indicators, data centralization and standardization are needed to solve the comparability between data indicators. After data centralization and standardization, the indicators of the original data are in the same order of magnitude, which is suitable for comprehensive comparative evaluation (CSDN, 2017).

Mean-centering involves subtracting variable averages from the original data. Since multivariate data is typically processed in table format (i.e. matrix) with columns as variables, mean-centering is usually known as column-centering (Sanchez, 2014). All we have to do with mean-centering is

to calculate the average value of each variable and subtract it from the data. It implies that each column will be converted in such a way that the mean of the resulting variable is zero.

Mean-centering can be done with the apply() function in R. In this case, the idea is to remove the mean in each column. This is done by declaring the apply() function and performing the mean-centering operation. Specify a vector of MARGIN, like c(1, 3, 4) you get the mean of the columns for each row and level of the third and fourth dimensions. The whole dataset Scans.arr is a four-dimensional array with the following dimensions:

- [1200 (points in time), 116 (brain regions), 4 (scans), 820 (people)]

The time series ‘mean.w’ corresponds to the average work of the whole brain during the time of the fMRI scan. This might reflect the general life processes during that time. Use the apply() function to subtract the mean from each column. Then use the original full data Scan.arr minus mean.w, the centered data Scan.cent could be achieved. Hence Scan.cent describes the remaining activity after those general life processes were subtracted.

The example plot below takes 200 points of the first brain regions in one scan of one person as the horizontal axis (index from 1 to 200). The colors of the lines correspond to the following:

- Black line: Scan1 (a subset of the original data)
- Red line: Scan1.cent (a subset of the centered data)
- Green line: Mean.w (a subset of the mean on each column)

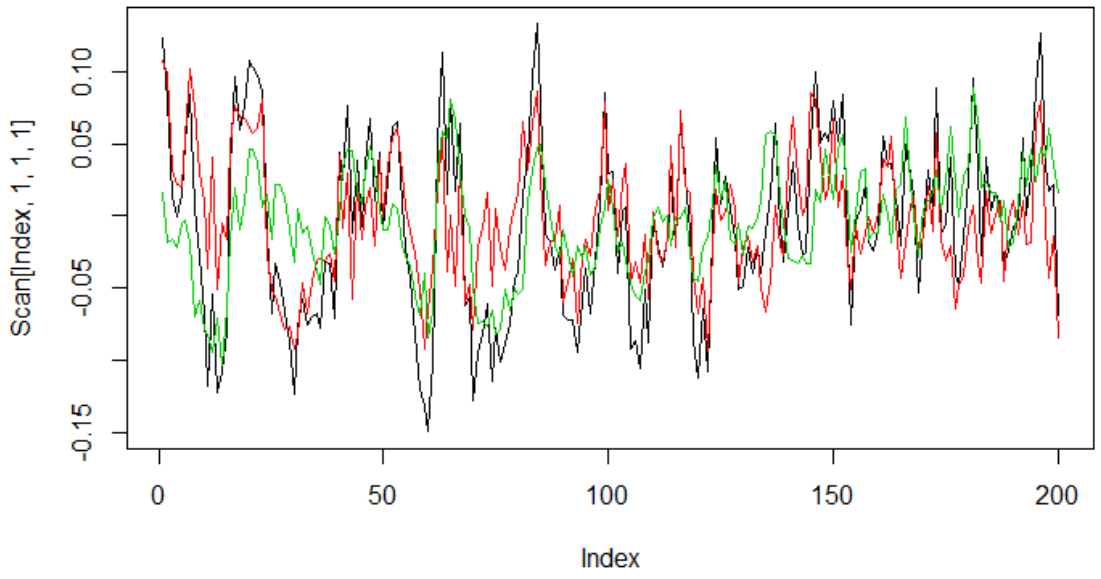


Figure 15. Example plot (three lines)

4.2 Check the Correlation between Variables

Compute a correlation matrix that is used to investigate the dependence between multiple variables at the same time. The result is a table containing the correlation coefficients between each variable and the others. Also, the histogram of the correlation matrix can clearly show the frequency of the coefficient values, and can clearly provide whether the variables have a positive or negative correlation.

For the centered data Scan.cent, the histogram of the correlation matrix (Figure 16) looks symmetrical when the correlation coefficient value is 0. It looks normal, but the tail of the distribution extends a little further to the right than it does to the left. It has both positive and negative correlations between the variables.

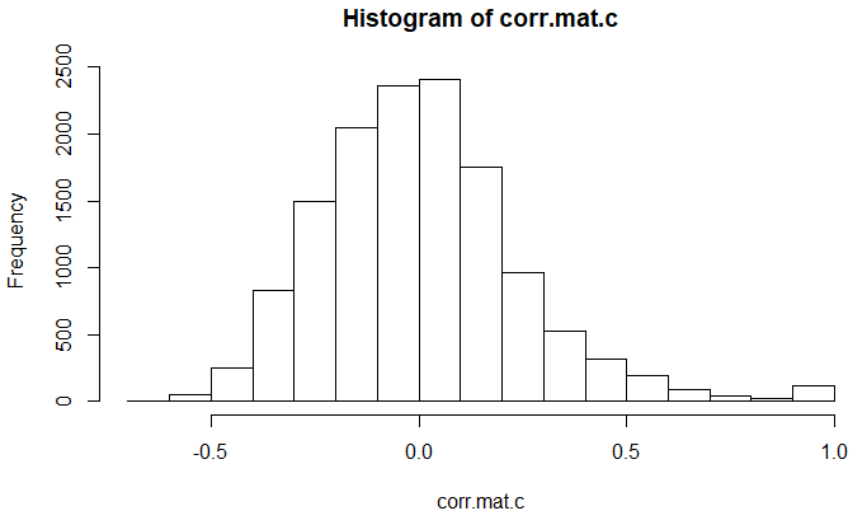


Figure 16. Histogram of correlation matrix (centered data)

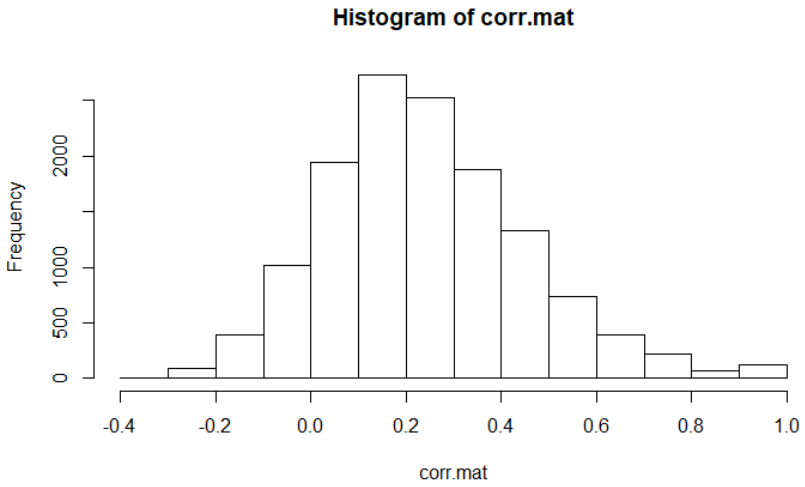


Figure 17. Histogram of correlation matrix (original data)

Based on the plot, it shows most correlation coefficient values are greater than 0. It shows some positive correlations between the variables in the original data. It means that as one variable increases, the other one tends to increase as well.

4.3 Calculate a Difference of a Series

4.3.1 Diff() Function

Calculating a difference of a time series could use diff() function in R. A common operation of time series, usually on non-stationary time series, is to get a difference of the series. A series of differences is an application that recursively calls the difference function n times. A simple way to look at an individual (first-order) difference is to treat them as x (t) - x (t - k) (DataCamp, 2018). Here, k is the number of lags to go back. The high-order difference is only a reapplication of each previous result difference. In R, the diff() function takes two notable arguments. The first one is the lag that is the number of periods. The second is differences, which is the order of difference.

For both the original data and the centered data, take differences of the k_{th} order as in diff(arr, k=k). For each dataset, consider 5 cases with k=0, 1, …, 4, where k=0 is the original data.

4.3.2 ACF Plot Analysis

The dashed horizontal lines in the plots are intended to give critical values for testing whether or not the autocorrelation coefficients are significantly different from zero. These limits are based on the approximate large sample standard error that applies to a white noise process, namely.

For the centered data:

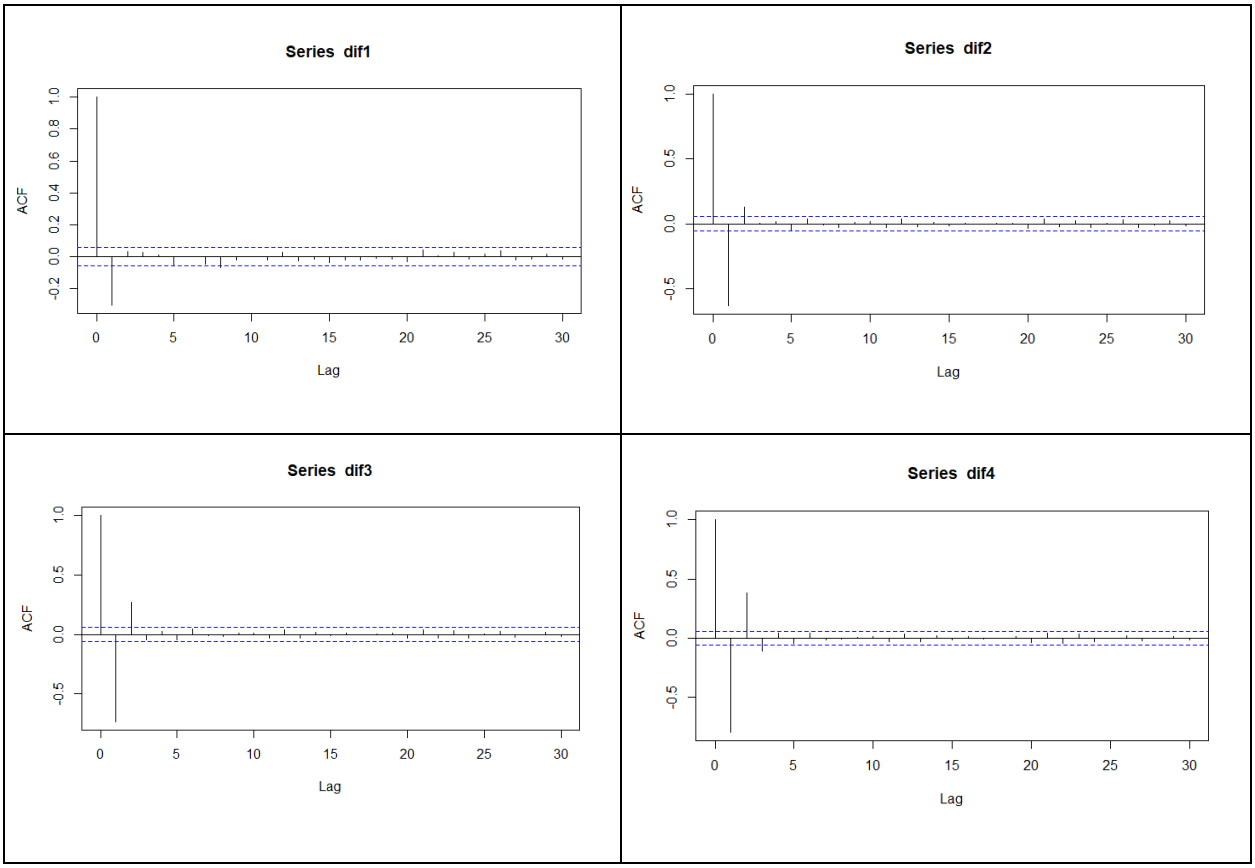


Figure 18. ACF plots (centered data, k=1,2,3,4)

The ACF plots above are the sample estimate of the autocorrelation function of 1-4st differences for the centered data. The above ACF plots note the alternating and tapering pattern.

Dif1 plot: Notice that the sample ACF values exceed these rough critical values at lags 1 and 8. This looks like it has negative lag 1 and lag 8 autocorrelations. But at 8 lag the significance is not very strong.

Dif2 plot: The sample ACF values exceed these rough critical values at lags 1 and 2. Lag 2 became significant, and lag 8 became not significant. It has an ACF which is near 0 for lags over 2 as predicted theoretically.

Dif3 plot: The sample ACF values also exceed the rough critical values at lags 1 and 2. But lag 2 is a little more significant that it was in Dif2 plot.

Dif4 plot: The sample ACF values also exceed the rough critical values at lags 1 and 2. Besides, lag 3 became significant, and lag 2 is still more significant than it was in Dif3 plot.

For the original data:

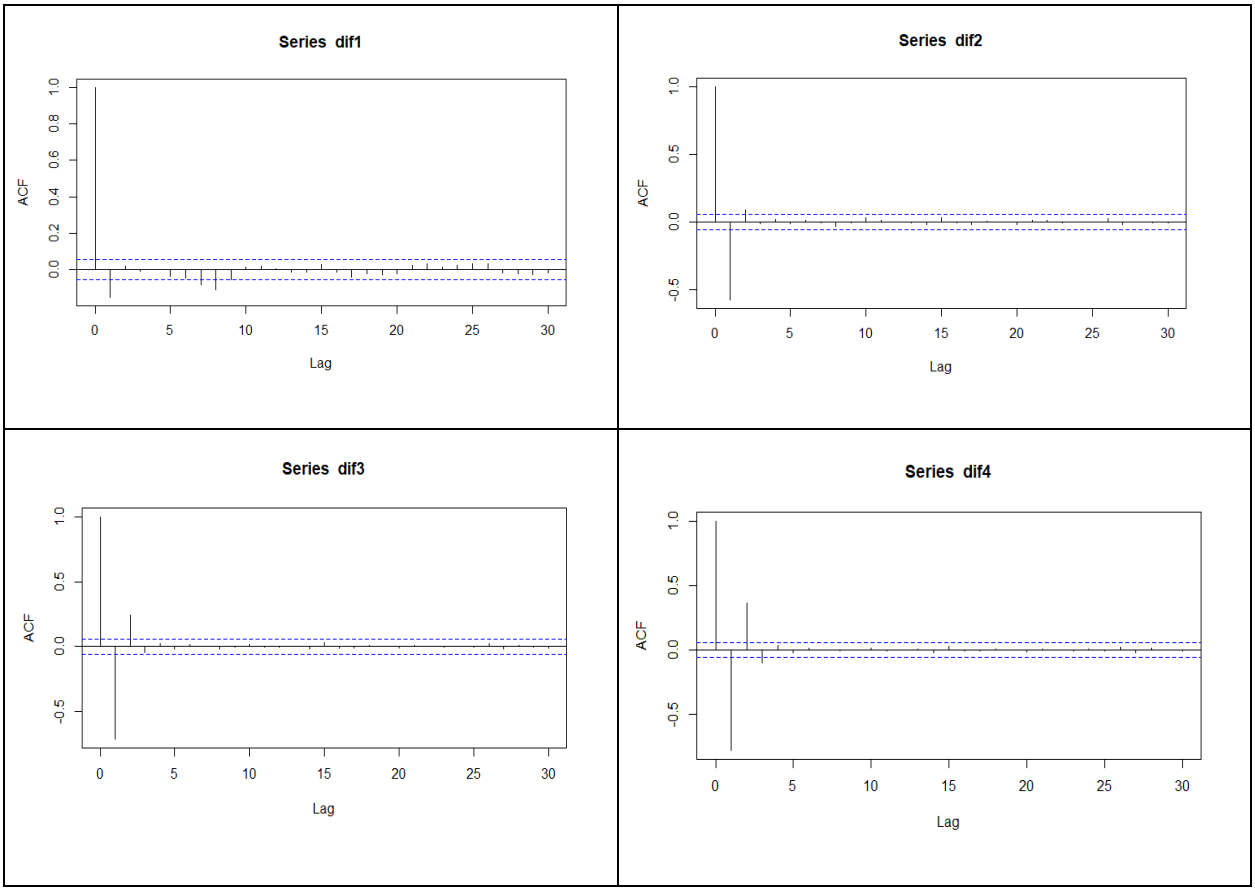


Figure 19. ACF plots (original data, k=1,2,3,4)

The above plots are the sample estimate of the autocorrelation function of 1-4st differences for the original data. The above ACF plots note the alternating and tapering pattern.

Dif1 plot: Notice that the sample ACF values exceed these rough critical values at lags 1, 7 and 8. This seems that it has negative lag 1, lag 7 and lag 8 autocorrelations. But at lag 7 and lag 8, the significance is not very strong.

Dif2: The sample ACF values exceed the rough critical values at lags 1 and 2. But, lag 7 and lag 8 became not significant. It has an ACF which is near 0 for lags over 2 as predicted theoretically.

Dif3: The sample ACF values exceed the rough critical values at lags 1 and 2, but lag 2 is a little more significant than it was in Dif2 plot.

Dif4: The sample ACF values exceed the rough critical values at lags 1, 2 and 3. Also, lag 2 and lag 3 are more significant than these were in Dif3 plot.

4.4 Correlations Consistent Check

4.4.1 Calculate the Correlations

Calculate correlations through the R code as follows, where Scans.arr is either the original data or differences, depending on which case are calculated. corr.arr should have dimensions: 116 x 116 x 4 (scans) x 820 (people).

```
corr.arr <- apply(Scans.arr, 3:4, cor)
corr.arr <- array(corr.arr,c(116,116,4,820))
dim(corr.arr)
[1] 116 116  4  820
```

The total number of calculations is 10 cases, including the original data and the differences with k=1,2,3,4 (5 cases), as well as the centering data and its differences with k=1,2,3,4 (5 cases).

If taking the correlations Y = corr.arr[1,2,,], it will have 4 values (from 4 scans) for each of the 820 people. Individual differences between people are likely to cause brain activity to be very

different. However, multiple scans on the same person are likely to be consistent. To verify this assumption, the study of checking the consistency of correlations for different scans of the same person could be carried out. In order to assess that, take Y as a response variable and fit a random effect model with a person as a factor X. Then the percent p of variability in Y explained by X could be calculated. The value of p close to 100% means that scans are very similar to each other. Hence, a large value of p is looked for.

4.4.2 Fit a Random Effect Model

The completely randomized design with a random effect assumes the following model.

$$y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

$$j = 1,2,\dots,n_i;\ i = 1,2,\dots,k;\ where\ \alpha_i \sim iidN(0,\sigma_\alpha^2)\ and\ \epsilon_{ij} \sim iidN(0,\sigma_\epsilon^2)$$

$$with\ \{\alpha_i\}\ independent\ of\ \{\epsilon_{ij}\}$$

If the data is perfectly balanced with equal numbers of individuals in each group (n_i = n for all i), it is possible to make modifications to a fixed effect ANOVA analysis by computing a different expected mean square (EMS) for inference (Larget, 2007).

First set the person from 1 to 820 as a factor X as the only independent variable. And the correlation Y which already calculated in previous steps as the dependent variable. Then use linear regression models to build the random effect model.

4.4.3 Calculate the Percent p of Variability in Y

In simple linear regression, r^2 is often called the coefficient of determination, because it is equal to the proportion of variability in Y (outcome variable) which is determined by the linear

relationship between X and Y. So here the value of p is equal to r^2. The summary function in R can be used to conduct r^2 after fitting the model.

```
model <- lm( y ~ x , data=df )
summary(model)$r.squared
[1] 0.6662254
```

4.4.4 Find a Better Value of p (r squared)

- Top 10 largest p

There are 116*115/2 cases of correlations. Because there are a large amount of cases of Y variables, a larger value of p should be found. Use a nested loop to calculate all the value of p over through all the 6670 cases and pick the top ten. The R loop code example is as follows:

```
res<-vector("numeric")
res<-matrix(nrow = 116, ncol = 116)
# generate a factor x from 1 to 820 as the person variable
x<-as.factor(rep(1:820,rep(4,820)))
for (i in 1:115) {
  for (j in ((i+1):116)) {
    Y<-corr.arr.o[i,j,,]    # 116*115/2 total cases of correlations
    y<-as.vector(Y)
    m<-lm( y ~ x )          # fit a model
    p<-summary(m)$r.squared # calculate the value of p
    res[i,j]<-p             # save all the p
  }
}
Res
# pick 10 largest values of p
res[order(res, decreasing=TRUE)][1:10]

[1] 0.7937858 0.7614425 0.7570771 0.7549371 0.7504789 0.7496170 0.7437273 0.7
410131 0.7406593 0.7398214
```

The result shown above is just in one data case (the original data). After that, the 10 data cases need to be compared, among them, find which could give us better values of p overall. The 10 cases include the original data and its differences with k=1,2,3,4 (5 cases), as well as the centered data and its differences with k=1,2,3,4 (5 cases). Details are shown in Table 10.

Case1: original data, k=0	Case6: centered data, k=0
Case2: original data, k=1	Case7: centered data, k=1
Case3: original data, k=2	Case8: centered data, k=2
Case4: original data, k=3	Case9: centered data, k=3
Case5: original data, k=4	Case10: centered data, k=4

Table 10. Ten data case information details

There are many ways to compare the two cases. For example, take differences between values of p, and check if the overall distribution (over 116*115/2 cases of correlations) tends to be more positive or negative and by how much.

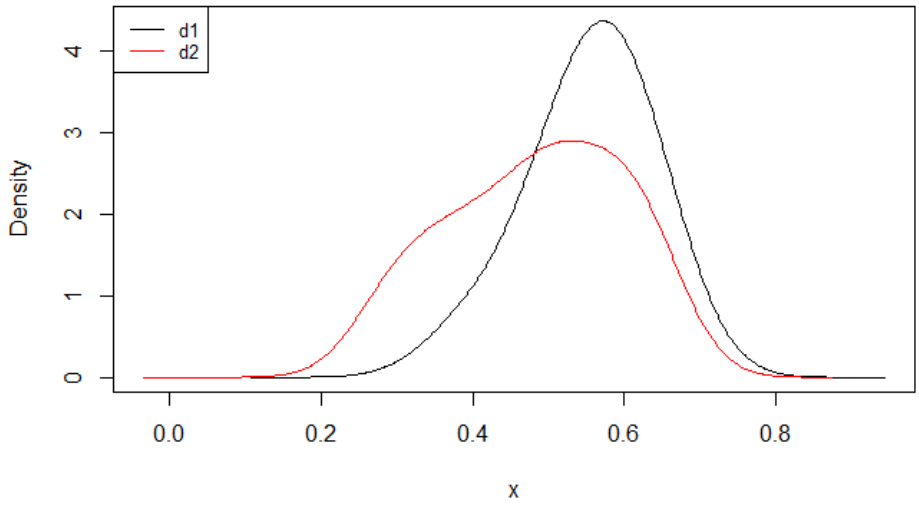


Figure 20. Comparison distribution between case1 and case2 (d1 vs.d2)

Based on the plot, the distribution 1 (d1) tends to be more positive than distribution 2 (d2). The following plot shows the distribution of delta_p (the difference between the two cases, res1-res2). The difference tends to be positive can be seen.

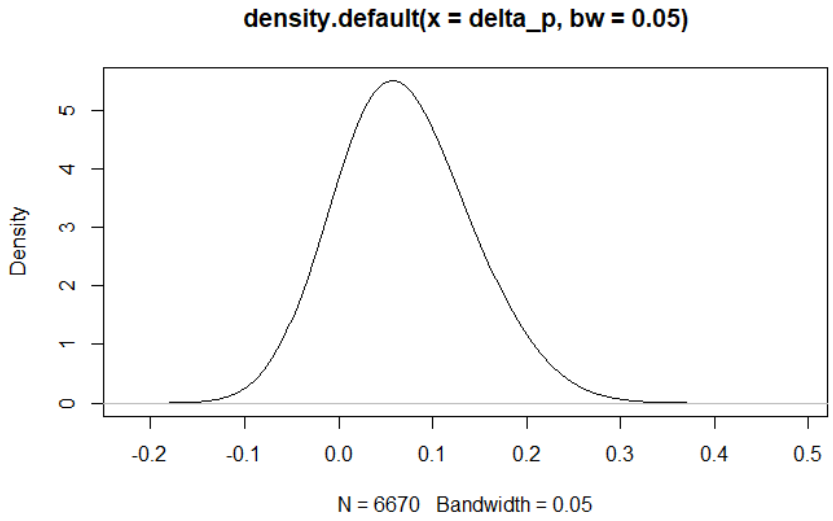


Figure 21. The difference between two cases (delta_p=res1-res2)

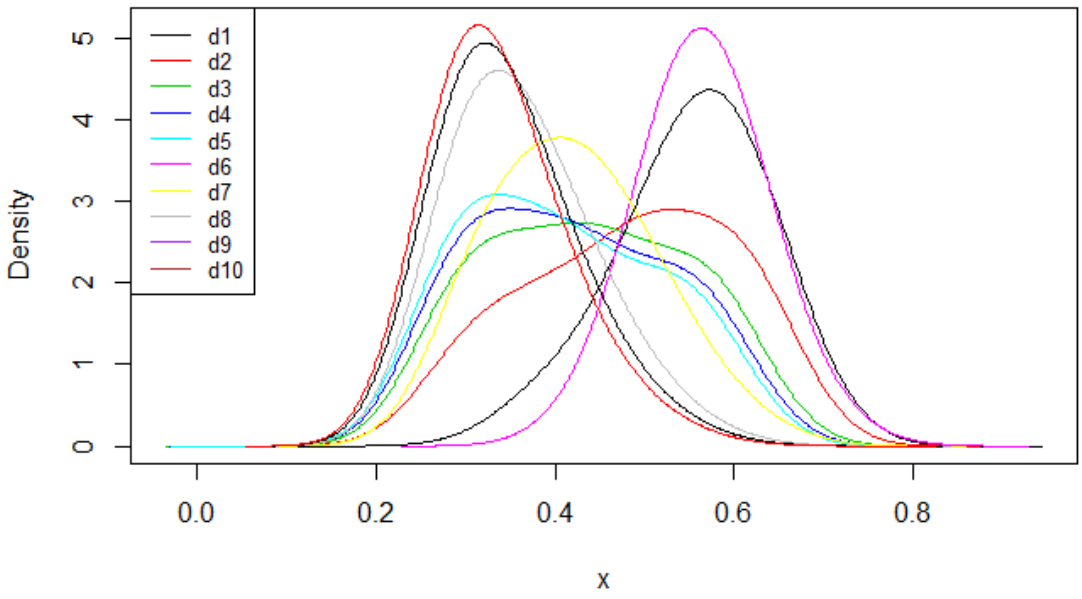


Figure 22. Ten distributions of ten data cases (d1~d10)

Figure 22 shows the ten distributions of ten data cases. It shows that d1 (case1 original data, k=0) might have larger values of p among all the distributions, and d10 (case10 centered data, k=4)

might have the smallest. To compare the difference between values of p among overall distributions, d10 would be chosen as the base.

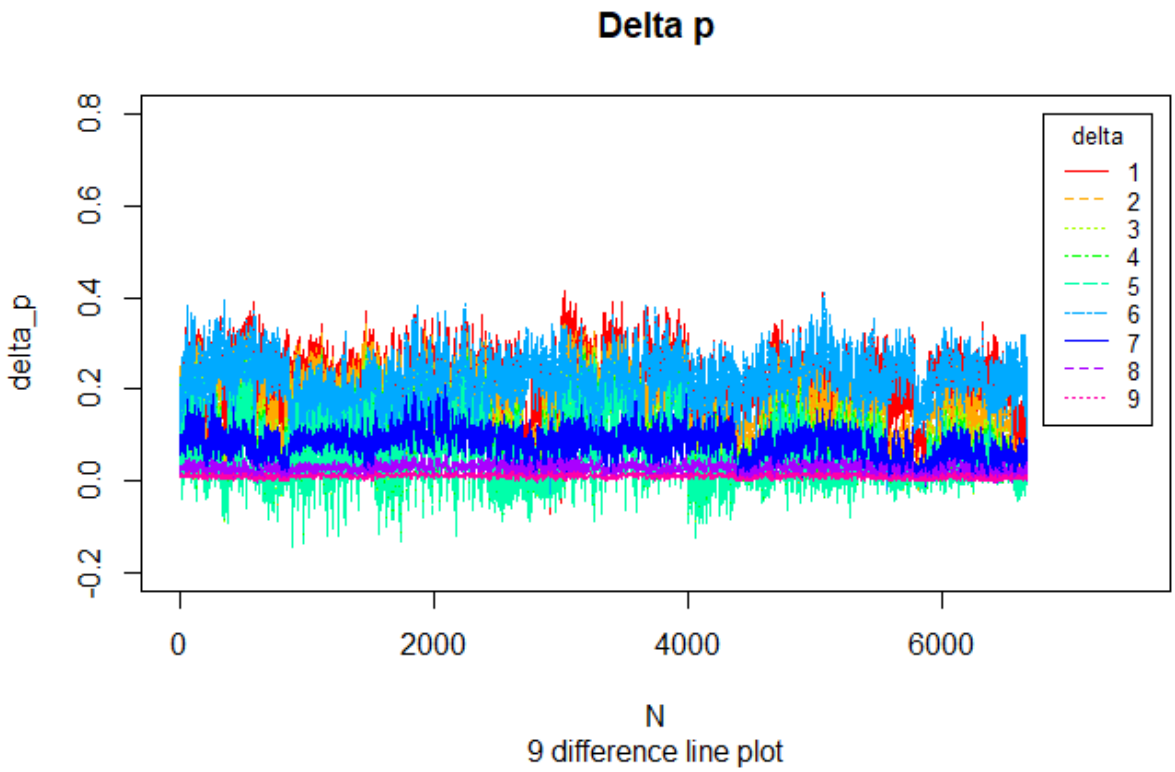


Figure 23. Delta_p plot (res10 as the base)

Based on the Delta p plot above and 9 previous density plots, most distributions trend to be positive. Delta1 and Delta6 tend to be more positive among all the distributions, and Delta5 tends to be more negative.

- Top 20~50 largest values of p

Start with the top 20 or 50 values of "res1" and then check the corresponding values for all 10 cases for the same 20 or 50 variables. Create a function to calculate 20 or 50 largest values of p for each data case (ten groups). Then create a line chart and combine the ten groups of the largest values.

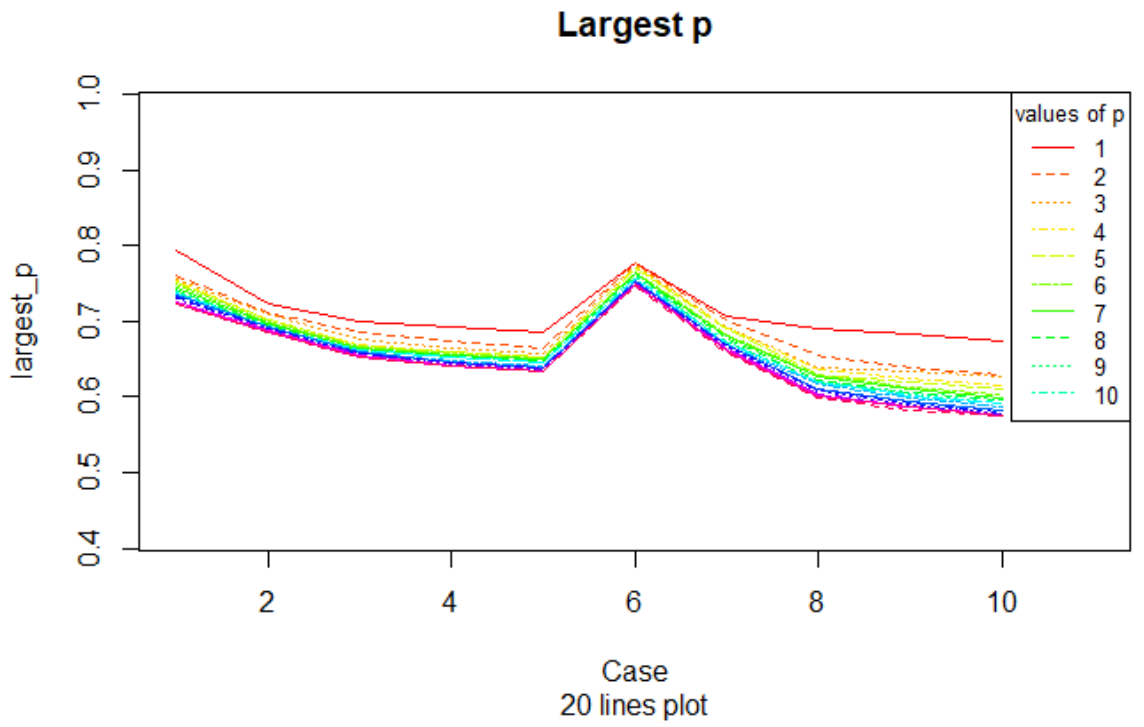


Figure 24. 20 largest values of p

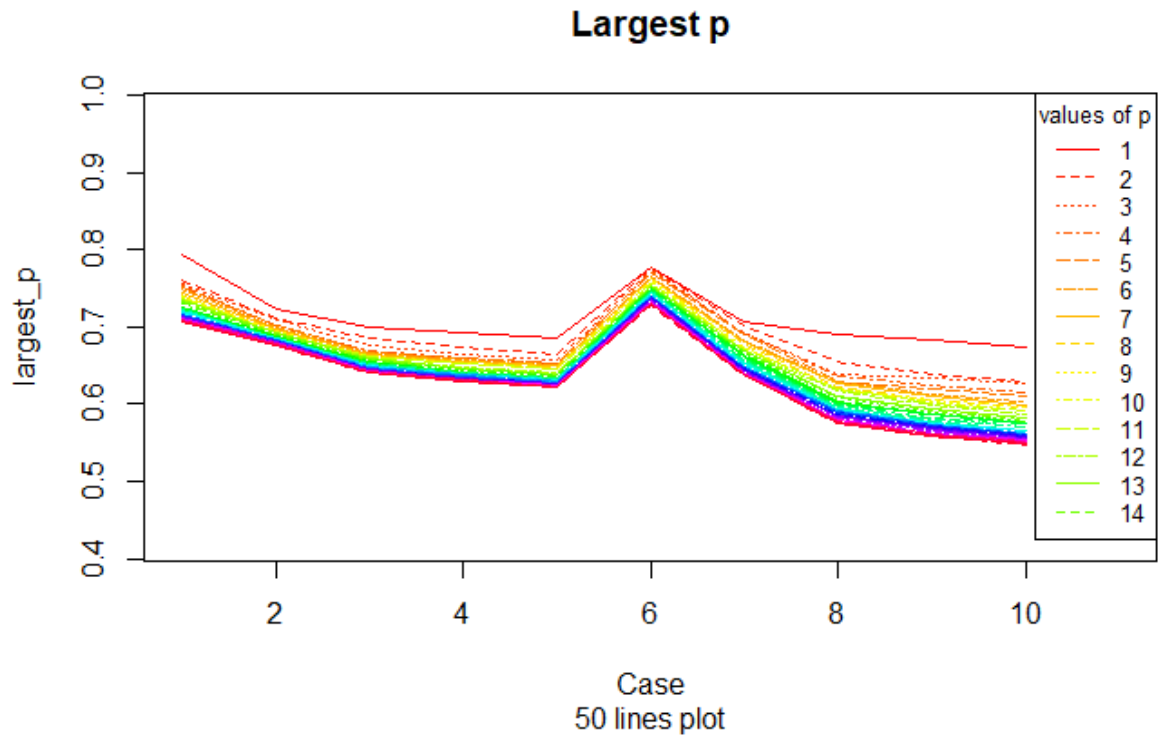
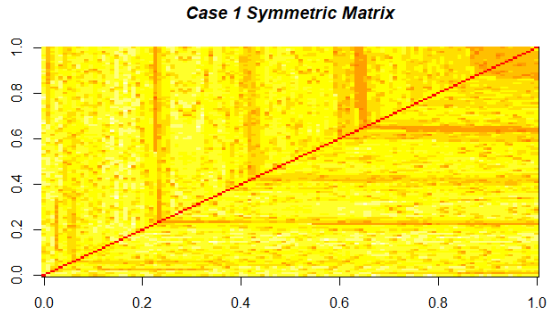


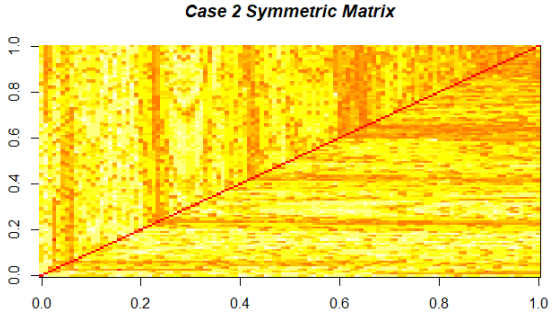
Figure 25. 50 largest values of p

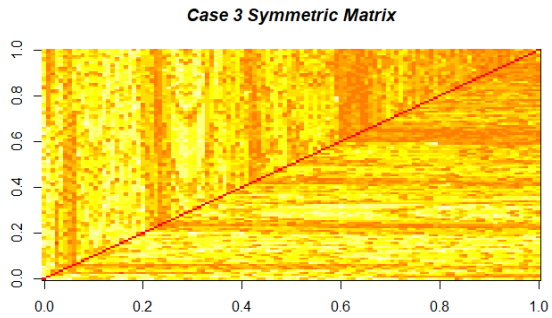
Based on the two plots above (Figure 24, 25), by comparing the ten cases, Case6 contains more large values of p which are more concentrated as well. On the contrary, the values of p in the 10^{th} case are the smallest and the most dispersed. So the largest case of p values should be Case6.

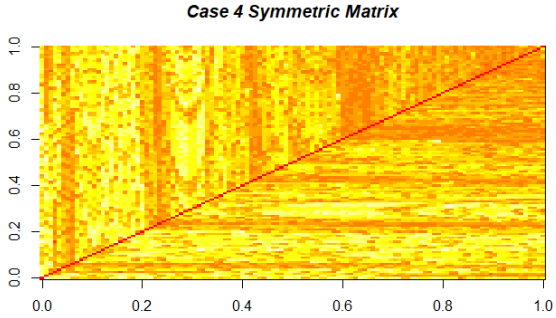
- Symmetric plot

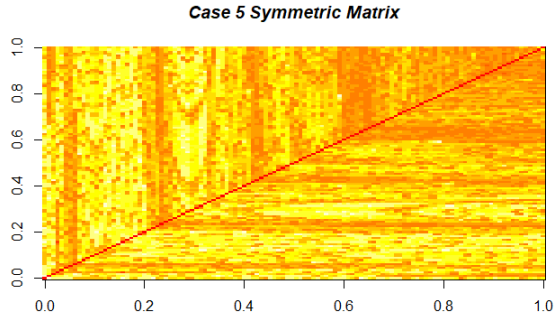
For each case, the 116*115/2 values of p can be organized into a symmetric 116 by 116 matrix. Then use image() function to plot a color image to represent those values. There are total ten color plots (case1~case10).

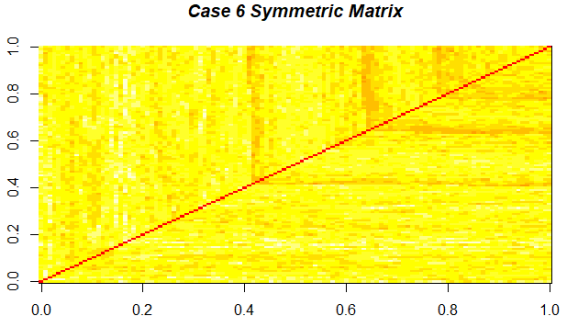












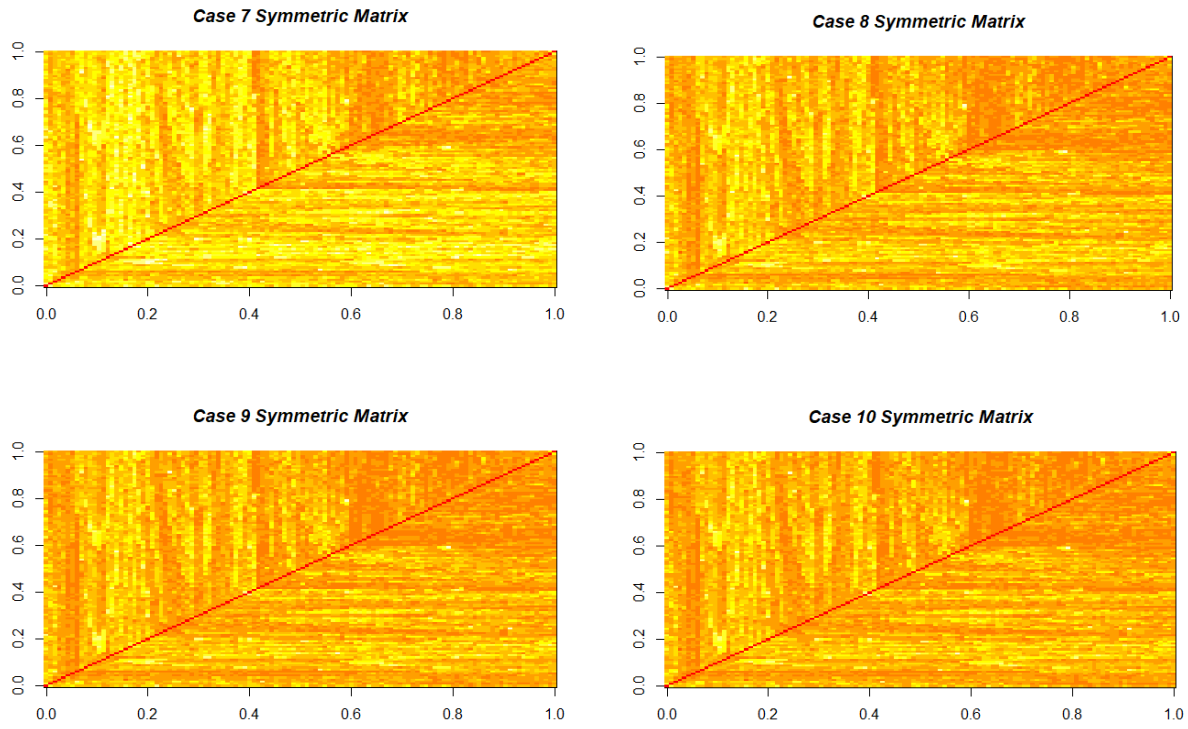


Figure 26. Symmetric matrix (case1~case10)

By comparing the ten plots above, Case6 has the lightest color, and the overall color distribution looks relatively uniform. However, Case9 and Case10 have the darkest overall color. Since the color intensity increases from red to white (dark to light), it shows Case6 contains the most number of large values of p.

5 Conclusions

One of the main purposes of this study is to reduce the autocorrelation of residuals by choosing the best smoothing method. It is found that the time series of human brain activity processed directly by smoothing splines have some problems including over-fitting and a large number of autocorrelations. To solve these issues, B-spline is selected for the curve smoothing in this research. The optimal smoothness is found by choosing the right knots and adjusting the number of knots. In the process of selecting knots, a set of knots is preliminarily screened out by

determining all the peaks and troughs (i.e. maximum and minimum) of the curve. Next, by specifying a fixed interval apart to filter the knots, the scope can be narrowed so that the knots do not get too close to each other. Then, the knots with the highest reduction in correlation are removed for further reduction. As a result, it was finally found that when the knots were at least 3 intervals apart, the DW value was closest to 2 and the correlation value was smallest.

In addition, this paper also finds that multiple fMRI scans of the same person also have correlations. And these correlations are kind of consistent. The study evaluates this by fitting a random effect model with people as a factor and correlation as a response variable. By calculating r-square, which is the percentage of variation in Y, whether the scans are similar to each other can be found. Finally, the study shows that the correlation is indeed consistent, and the correlation is the most consistent in the centered data with 0^{th} order.

6 Limitation and Future Researches

In smoothing technology, this study only used B-spline as a smoothing curve method. In future research, the advantages of the knot insertion technique and curve fitting method could be combined more reasonably and effectively.

In the aspect of the knot insertion algorithm, this research chooses the drastically changing area in the data as knots, such as the location of maximum and minimum values. Other methods of knot selection can also be considered. For example, setting more knots in the place with complex data changes; on the contrary, fewer knots will be selected in place that seems more stable. Beside, cross-validation is also a good method. A study also could be conducted by comparing the efficiency and accuracy of these different methods.

Appendix

Appendix 1. Select a set of knots at least 3 intervals apart and analysis plot

---

```
load(“C:/Users/Subset.Smooth.arr”)

dim(Subset.Smooth.arr)

load("C:/Users/Subset.Scans.arr")

dim(Subset.Scans.arr)

Subset.Resid.arr <- Subset.Scans.arr - Subset.Smooth.arr

Index <- 1:1200

plot(Index,Subset.Resid.arr[,1,1,1],main="Residuals from Smoothing")

ss<-Subset.Scans.arr[,1,1,1]


# Identify knots at all the maximum and minimum points through all
observations

i<-2:1200

location<-as.numeric(which((ss[i-1]<ss[i] & ss[i+1]<ss[i])|(ss[i-1]>ss[i] &
ss[i+1]>ss[i])))

length(location)


# create a data frame included knots locations and values

y.value<-ss[location]

set.knots<-data.frame(location, y.value)

colnames(set.knots) <- c("location","y.value")


# start a loop to select knots

count=0

n=3     # at least 3 intervals apart

for (j in 1:(nrow(set.knots)-1)){

  j=j-count

  #the knots should be at least 3 intervals apart

  #the minima or maxima location minus the previous one is greater than 3

  l = set.knots$location[j+1]-set.knots$location[j]

  if (l<n){
```

```
    #delete the knot which is too close to the previous one
    set.knots=set.knots[-(j+1),]
    count=count+1
  }
  else{
    set.knots=set.knots
  }
}
head(set.knots)


require(splines)
k.loc<-set.knots$location
regr.spline <- lm(ss ~ bs(Index, df = NULL, knots=k.loc, degree = 3,
intercept=T))
fits.bs.mm <- regr.spline$fitted # from spline model
res<-Subset.Scans.arr[,1,1,1]-fits.bs.mm


# D-W test
# library(lmtest)
dwtest(res~Index, alt="two.sided")
cor(res[-1], res[-length(res)])


#count number of knots
length(set.knots$location)


# fitting models with splines
# plotting the data with the regression spline overlain:
x.values <- seq(from=min(Index), to=max(Index), length=1200)
plot(Index, ss); lines(x.values, predict(regr.spline,
data.frame(x=x.values)),col='red')


# the baseline of the correlation value
cor.baseline<-cor(res[-1], res[-length(res)])
cor.baseline
```

Appendix 2. Further reduction and autocorrelation check

```
#check how much the correlation between consecutive residuals is reduced
correlation<-c()
reduction<-c()


for (k in 1:length(k.loc)) {
  regr.spline <- lm(ss ~ bs(Index, df = NULL, knots=k.loc[-k], degree = 3,
intercept=T))
  fits.bs.mm <- regr.spline$fitted
  res<-Subset.Scans.arr[,1,1,1]-fits.bs.mm
  correlation[k]<-cor(res[-1], res[-length(res)])
  # Compare the current correlation to the baseline, get the reduction
  reduction[k]<-cor.baseline - correlation[k]
}
correlation.a<-data.frame(k.loc, correlation, reduction)

# searching for a positive or the largest reduction
subset(correlation.a, reduction > 0)
cor.max<-which.max(correlation.a$reduction)
cor.max
value<-correlation.a[cor.max,]
value

# delete the knot which gives the max reduction
k.loc<-k.loc[-cor.max]
```

Appendix 3. Calculate the largest 10 value of p (R-square)

```
# To increase the storage capacity
memory.limit(size=56000)
# load in the original data
load("C:/Users/Scans.arr")
dim(Scans.arr)
corr.arr<-apply(Scans.arr, 3:4, cor)
dim(corr.arr)
corr.arr<-array(corr.arr, c(116,116,4,820))
dim(corr.arr)

res<-vector("numeric")
res<-matrix(nrow = 116, ncol = 116)
x<-as.factor(rep(1:820,rep(4,820)))
Sys.time()
for (i in 1:115) {
  for (j in ((i+1):116)) {
    Y<-corr.arr[i,j,,]
    y<-as.vector(Y)
    m<-lm( y ~ x )
    p<-summary(m)$r.squared
    res[i,j]<-p
  }
}
Sys.time()
res

# 10 largest values
v <- res[order(res, decreasing=TRUE)][1:10]
# find the posision of 10 largest values
p <- which(res>=sort(res, decreasing = T)[10], arr.ind = T)
# determine the order of the 10 largest values in decreasing order
v.order <- order(res[p], decreasing = T)
p[v.order, ]
```

Appendix 4. Comparison plot (d1~d10)

```
d1 <- (density(res1, bw = 0.05))
d2 <- (density(res2, bw = 0.05))
d3 <- (density(res3, bw = 0.05))
d4 <- (density(res4, bw = 0.05))
d5 <- (density(res5, bw = 0.05))
d6 <- (density(res6, bw = 0.05))
d7 <- (density(res7, bw = 0.05))
d8 <- (density(res8, bw = 0.05))
d9 <- (density(res9, bw = 0.05))
d10 <- (density(res10, bw = 0.05))
plot(range(d1$x, d2$x, d3$x, d4$x, d5$x, d6$x, d7$x, d8$x, d9$x, d10$x),
     range(d1$y, d2$y, d3$y, d4$y, d5$y, d6$y, d7$y, d8$y, d9$y, d10$y),
     col=c(1:8,"purple","brown"), type = "n", xlab = "x", ylab = "Density")
lines(d1, col = 1)
lines(d2, col = 2)
lines(d3, col = 3)
lines(d4, col = 4)
lines(d5, col = 5)
lines(d6, col = 6)
lines(d7, col = 7)
lines(d8, col = 8)
lines(d9, col = 9)
lines(d10, col = 10)
legend("topleft",
legend=c("d1","d2","d3","d4","d5","d6","d7","d8","d9","d10"),
       col=c(1:8, "purple","brown"), lty=1, cex=0.8)
```

Appendix 5. Comparing plot (largest 20 or 50 values)

```
# 20 largest values
largest20 <- function(X){
  largest <- X[order(X, decreasing=TRUE)][1:20]
  largest
}
#largest20(res1)
# Create Line Chart
largest_list<-
list(largest20(res1),largest20(res2),largest20(res3),largest20(res4),

largest20(res5),largest20(res6),largest20(res7),largest20(res8),
                    largest20(res9),largest20(res10))
largest_p <- do.call(cbind, largest_list)
xrange20 <- range(c(1:11))
yrange20 <- range(c(0.7:1))
# set up the plot
plot(xrange20, yrange20, type="n", xlab="Case",
     ylab="largest_p" )
colors <- rainbow(20)
linetype <- c(1:20)
# add lines
for (i in 1:20) {
  lines(c(1:10), largest_p[i,], type="l", lwd=1.5,
        lty=linetype[i], col=colors[i])
}
# add a title and subtitle
title("Largest p", "20 lines plot")
# add a legend
legend("topright", legend=c(1:10), cex=0.8, col=colors, lty=linetype,
title="values of p")
```

Appendix 6. Symmetric matrix plot

```
#make a symmetric 116 by 116 matrix
sym_mat1 <- matrix(rep(0,116*116), nrow=116)
sym_mat1[lower.tri(sym_mat1)] <- res1
#transpose matrix
sym_mat1 <- t(sym_mat1)
#populate the lower triangle, which was formerly the upper triangle
sym_mat1[lower.tri(sym_mat1)] <- res1
sym_mat1
#check symmetry using the isSymmetric() function
isSymmetric(sym_mat1)
image(sym_mat1, main = "Case 1 Symmetric Matrix", font.main = 4)
```

References

Bradley. (1968). Distribution-Free Statistical Tests, Chapter 12.

Carew, J.D., Wahba, G., Xie, X., Nordheim, E.V., Meyerand, M.E. (2002).Optimal spline smoothing of fMRI time series by generalized cross-validation. *Neuroimage, 18*, 950–961.

Chen, S. (2015). *Statistical methods to analyze massive high-dimensional neuroimaging data.* (Doctoral dissertation, The Johns Hopkins University)

CSDN.(2017, October 31). Centralization and Standardization. Retrieved from https://blog.csdn.net/qq_40245123/article/details/78399736

DataCamp.(2018) *Calculate a difference of a series using diff()*. DataCamp, Inc. Newyork, NY, USA, https://www.datacamp.com/.

Dung, V.T., Tjahjowidodo, T. (2017). A direct method to solve optimal knots of B-spline curves: An application for non-uniform B-spline curves fitting. PLoS ONE, 12(3): e0173857. https://doi.org/10.1371/journal.pone.0173857

Friston, K.J., Josephs, O., Zarahn, E., Holmes, A.P., Rouquette, S., Poline, J.-B. (2000). To smooth or not to smooth? *Neuroimage*, *12*, 196–208.

Larget, B. (2007, March 15). *Random Effects in R.* University of Wisconsin Madison.

NCSS Statistical Software. (2019). *X-bar Charts* (Chapter 244-1). NCSS, LLC. Kaysville, Utah, USA, ncss.com/software/ncss.

NIST/SEMATECH e-Handbook of Statistical Methods. (2012). Retrieved from http://www.itl.nist.gov/div898/handbook/.

PSU. (2018). Probability Theory and Mathematical Statistics. Retrieved from https://newonlinecourses.science.psu.edu/stat414/node/330/

Racine, J.S. (2018). A primer on regression splines. Retrieved from https://cran.r-project.org/web/packages/crs/vignettes/spline_primer.pdf

Sanchez, G.(2014, January 15). 6 way of mean-centering data in R. Retrieved from https://www.gastonsanchez.com/visually-enforced/how-to/2014/01/15/Center-data-in-R/

Smoothing. (2019). In Wikipedia. Retrieved June 12, 2019, from https://en.wikipedia.org/wiki/Smoothing

Talebitooti, R., Shojaeefard, M.H., Yarmohammadisatri, S. (2015). Shape design optimization of cylindrical tank using b-spline curves. *Computer & Fluids*, *109*, 100–112.