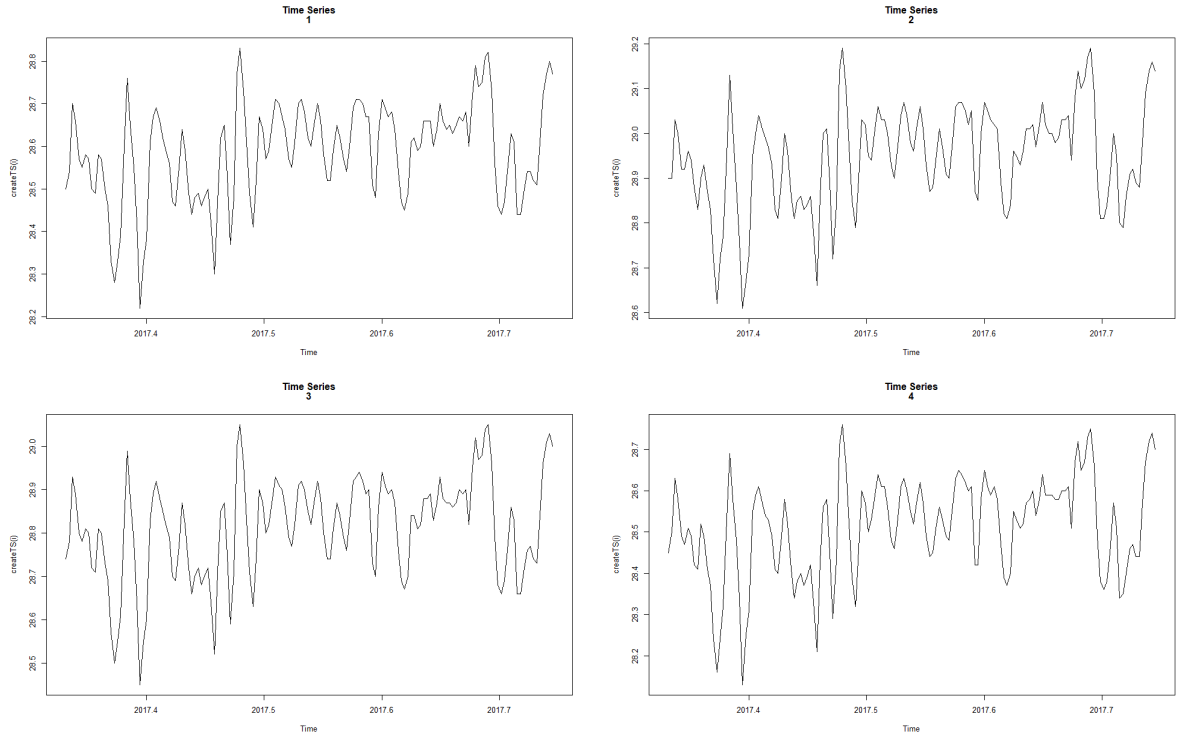
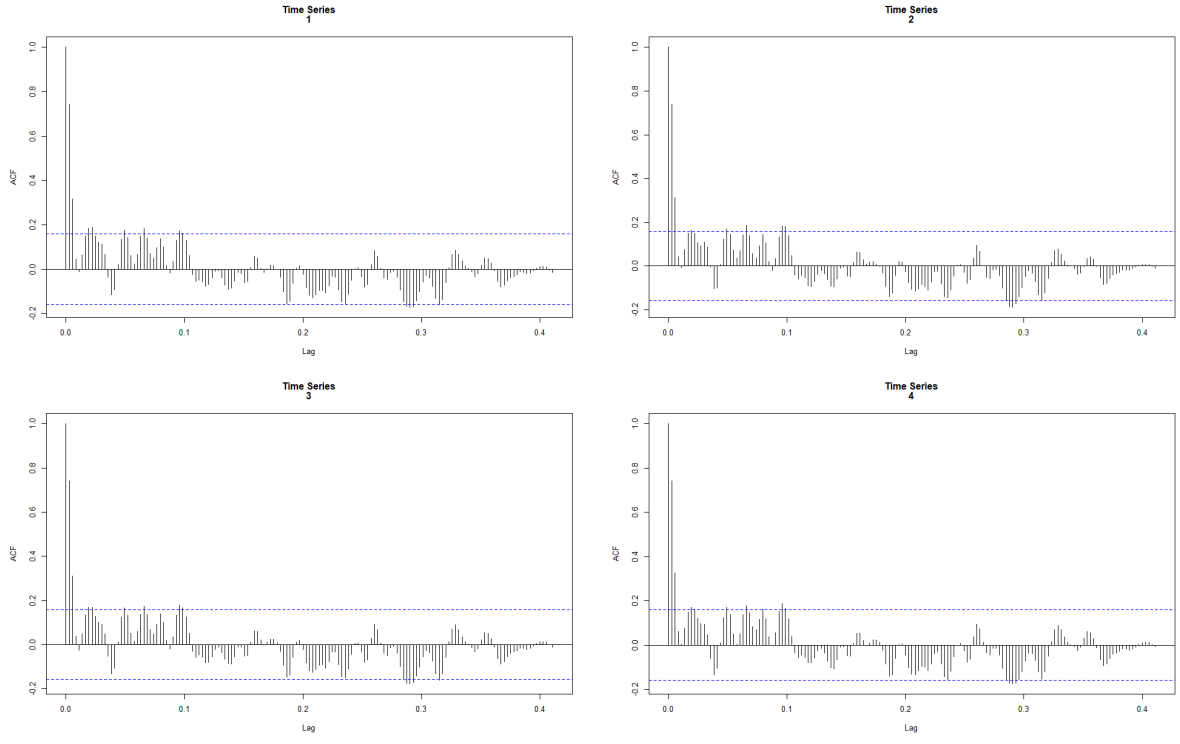
1. Linear model for temporal and spatial dependency

To have a rough look at the data, I first create the time series format for the data and plot them. As all stations show similar patterns, I only present the first four stations.



To further investigate the autocorrelation of the data, I also plot the autocorrelation of the for the original data with maximum lags.



As we can see, the first and second lags have the highest correlation whereas the rest of the lags do not pass the significant level. There is also damping effect, because the correlation shows a sequence of decreasing and increasing. This might be due to the oscillating nature of the data where it fluctuates in a specific pattern, but it has not passed the significant level so there is no need to consider. Thus, I will only consider 2 lags.

Let $Y_{t,j}$ be the pressure measured at time t and station j, t = 1, ..., 152 and j = 1, ..., 17. Define

$$Y_j = \begin{pmatrix} Y_{3,j} \\ Y_{4,j} \\ \vdots \\ Y_{152,j} \end{pmatrix}, lag1_j = \begin{pmatrix} Y_{1,j} \\ Y_{2,j} \\ \vdots \\ Y_{150,j} \end{pmatrix}, lag2_j = \begin{pmatrix} Y_{2,j} \\ Y_{3,j} \\ \vdots \\ Y_{151,j} \end{pmatrix}, for\ j = 1, \ldots, 17.$$

Then a simple linear model for j^{th} station is

$$\hat{Y}_j = \alpha_j + \beta_{1j} lag1_j + \beta_{2j} lag2_j\ (\boldsymbol{model1}),$$

assuming the response depends on lag1, lag2, and a constant α_j. Thus, fitting model 1 for all stations, we have 17 different estimates of $\alpha_j, \beta_{1j}\ and\ \beta_{2j}$.

```
print(c(summary(beta1),sd(beta1)))

##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## 1.11066162 1.13108405 1.14441906 1.14524697 1.15812741 1.18101774
##
## 0.01877344

print(c(summary(beta2),sd(beta2)))

##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
## -0.57845718 -0.54637324 -0.52876469 -0.52962182 -0.51153173 -0.47982377
##
##  0.02540246
```

We can see that $\beta_{1j}\ and\ \beta_{2j}$ do not vary much, so we can further assume $\beta_{1j} = \beta_1$ and $\beta_{2j} = \beta_2$ to form a single model for all stations. We can also assume α_j depends on spatial covariates, elevation, latitude and longitude. Thus, a linear model for α_j would be

$$\alpha_j = \gamma_0 + \gamma_1 elevation_j + \gamma_2 latitude_j + \gamma_3 longitude_j.$$

```
summary(lm.alpha)
## Call:
## lm(formula = alpha ~ nlat + elon + elev)
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.9822434  6.0217406   5.145 0.000188 ***
## nlat        -0.3130717  0.0332509  -9.415 3.60e-07 ***
## elon         0.0804398  0.0552782   1.455 0.169339
## elev        -0.0028173  0.0004459  -6.319 2.66e-05 ***
## Residual standard error: 0.04925 on 13 degrees of freedom
## Multiple R-squared:  0.974,  Adjusted R-squared:  0.968
## F-statistic: 162.5 on 3 and 13 DF,  p-value: 1.488e-10
```

As we can see, R-squared indicates a good fit and all predictors except for longitude indicate statistical significance.

Thus, to construct a single model for all stations, we can further plug the linear model for α_j in model1. Then the linear model for j^{th} station is

$$\hat{Y}_j = \gamma_0 + \beta_1 lag1_j + \beta_2 lag2_j + \gamma_1 elevation_j$$
$$+\gamma_2 latitude_j + \gamma_3 longitude_j \ (\boldsymbol{model2})^*,$$

assuming the response depends on lag1, lag2, and spatial data such as elevation, latitude and longitude.

```
summary(lm.fit)
## Call:
## lm(formula = Y ~ lag1 + lag2 + Elev + Nlat + Elon)
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.219e+01  7.659e-01  15.920   <2e-16 ***
## lag1         1.144e+00  1.691e-02  67.673   <2e-16 ***
## lag2        -5.279e-01  1.710e-02 -30.867   <2e-16 ***
## Elev        -1.230e-03  6.263e-05 -19.640   <2e-16 ***
## Nlat        -2.337e-03  3.657e-03  -0.639    0.523
## Elon         6.701e-03  6.081e-03   1.102    0.271
## Residual standard error: 0.06633 on 2544 degrees of freedom
## Multiple R-squared:  0.9258, Adjusted R-squared:  0.9257
## F-statistic:  6350 on 5 and 2544 DF,  p-value: < 2.2e-16
```

*Note: the model is fitted with a data that combines the responses, 2 lags, latitude, longitude and elevation for all 17 stations, effectively a 17*150 by 6 matrix. Therefore, all coefficient estimates are minimum squared error estimates.

As we can see, latitude and longitude do not show statistical significance, so we can simplify the model without losing accuracy by using

$$\hat{Y}_j = \gamma_0 + \beta_1 lag1_j + \beta_2 lag2_j + \gamma_1 elevation_j \ (\boldsymbol{model3}).$$

```
summary(lm.fit1)
## Call:
## lm(formula = Y ~ lag1 + lag2 + Elev)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.202569 -0.038137 -0.001095  0.043156  0.234051
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.146e+01  3.633e-01   31.55   <2e-16 ***
## lag1         1.144e+00  1.690e-02   67.70   <2e-16 ***
## lag2        -5.276e-01  1.710e-02  -30.86   <2e-16 ***
## Elev        -1.278e-03  4.534e-05  -28.18   <2e-16 ***
## Residual standard error: 0.06632 on 2546 degrees of freedom
## Multiple R-squared:  0.9258, Adjusted R-squared:  0.9257
## F-statistic: 1.059e+04 on 3 and 2546 DF,  p-value: < 2.2e-16
```

The output shows that removing latitude and longitude does not increase the residual standard error nor has much impact on R-squared so it's safe to simplify the model.

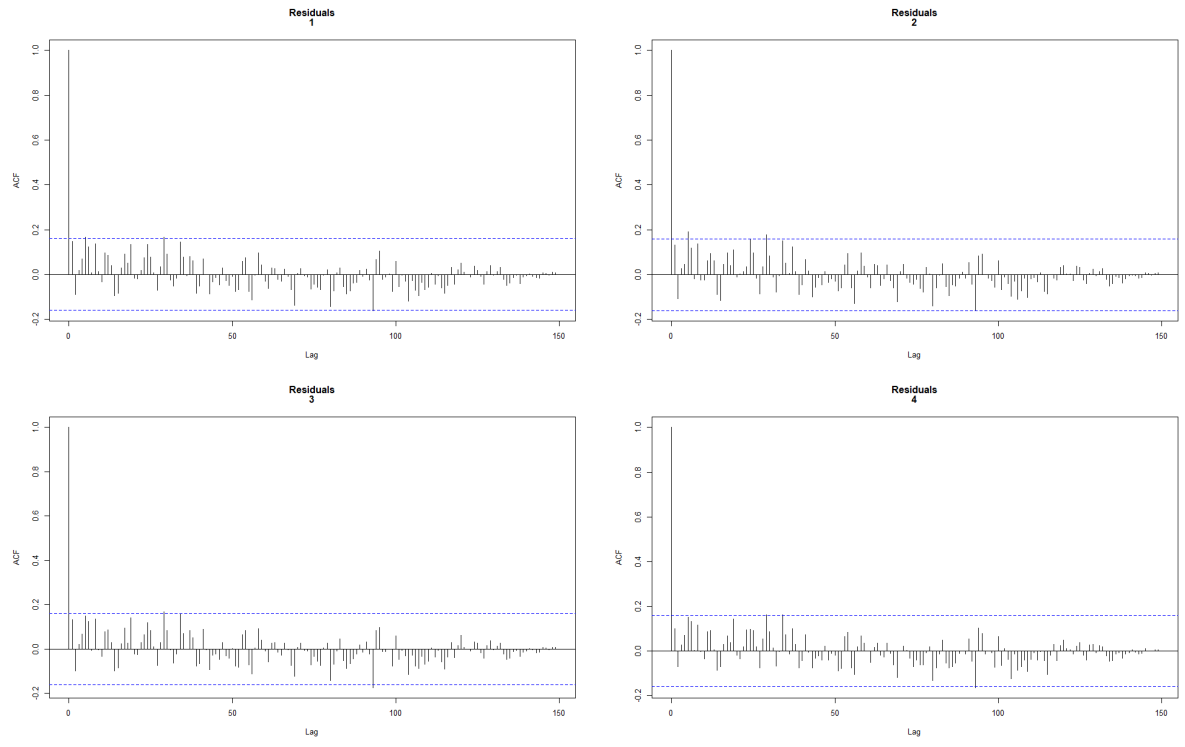
To consider interaction between lag1 and lag2, we need

$$\hat{Y}_j = \gamma_0 + \beta_1 lag1_j + \beta_2 lag2_j + \gamma_1 elevation_j + \gamma_2 lag1_j * lag2_j \ (\boldsymbol{model4}).$$

```
summary(lm.fit2)
## Call:
## lm(formula = Y ~ lag1 + lag2 + Elev + lag1 * lag2)
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.913e+01  1.524e+01   1.912   0.0560 .
## lag1         5.285e-01  5.314e-01   0.994   0.3201
## lag2        -1.144e+00  5.318e-01  -2.151   0.0316 *
## Elev        -1.289e-03  4.635e-05 -27.803   <2e-16 ***
## lag1:lag2    2.150e-02  1.854e-02   1.160   0.2463
## Residual standard error: 0.06631 on 2545 degrees of freedom
## Multiple R-squared:  0.9258, Adjusted R-squared:  0.9257
## F-statistic:  7941 on 4 and 2545 DF,  p-value: < 2.2e-16
```

Including the interaction term does not improve the model much so to simplify the model we can proceed with model3.

Thus, the residuals of the fitted model (i.e. model3) should have very low serial correlation.



As we can see, we have much lower serial correlation comparing to the autocorrelation plots of original data. However, the R-squared of model3 (0.9258) suggests that it's still far from a good fit model. This might be due to the simplicity of the linear model, and it only considers temporal dependency and spatial dependency but neglect the fact that stations

closer to each other might have higher correlations. Following will show how we can model this by assuming the MVN (multivariate normal) distribution of the standardised residuals.

2. MNV distribution of Standardised Residuals

Standardising the residuals is to cancel the noise and capture the correlation between each station.

Residuals for station j are defined as

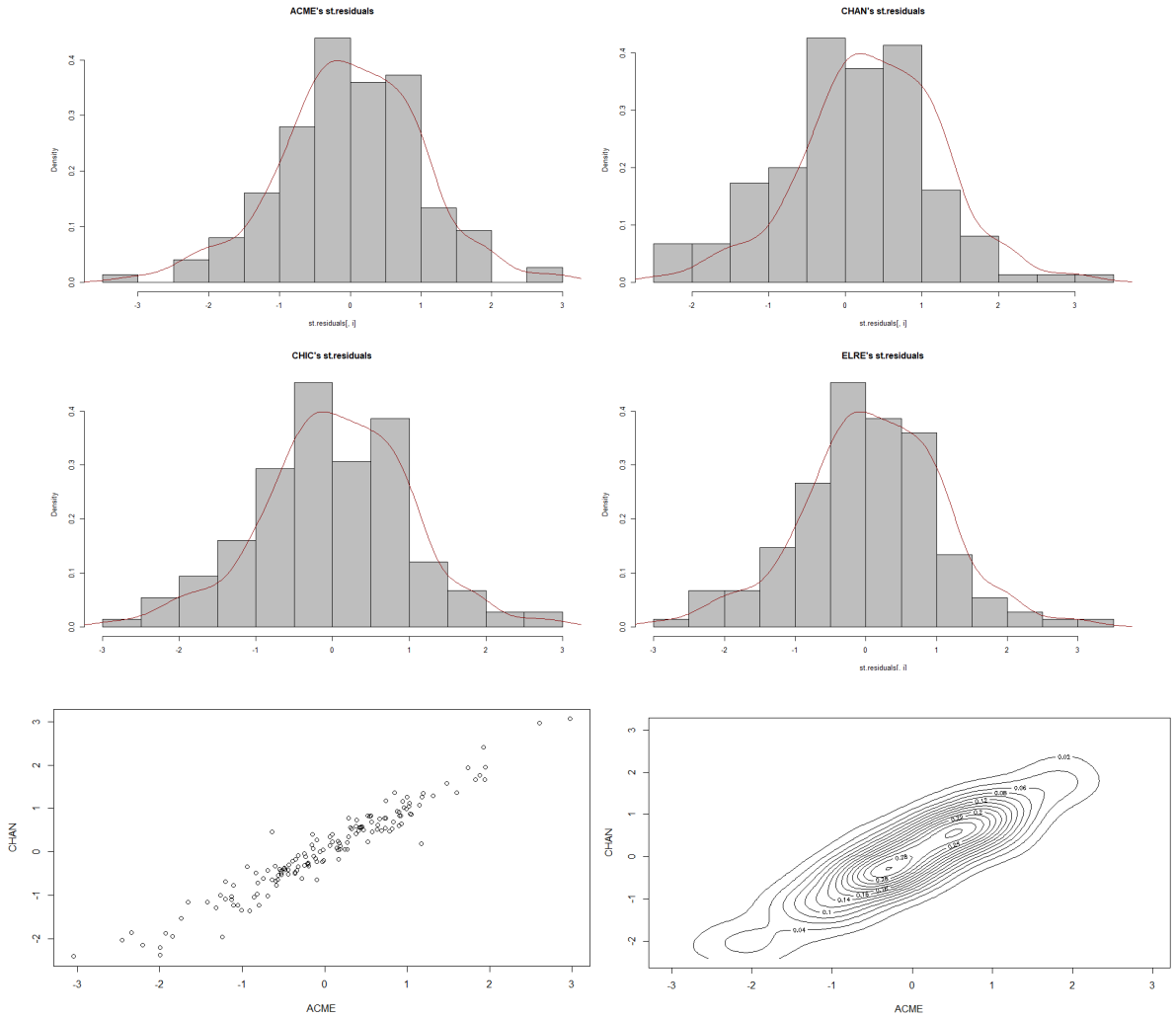
$$\varepsilon_j = Y_j - \hat{Y}_j = Y_j - (\gamma_0 + \beta_1 lag1_j + \beta_2 lag2_j + \gamma_1 elevation_j),$$

and the standardised residuals are calculated by

$$\varepsilon_j^* = \frac{\varepsilon_j}{standard\ deviation\ (\varepsilon_j)} = \frac{\varepsilon_j}{\sqrt{\sum_{i=1}^{150}(\varepsilon_{ij} - \overline{\varepsilon_j})^2/149}},$$

assuming residuals for each station follow normal distribution.

As we can see from the histograms of standardised residuals for the first four stations, it's safe to assume normally distributed standardised residuals for each station. Further we can assume standardised residuals from all stations are MVN distributed.



From the bivariate scatter and contour plots for the first two stations, we can see that there is higher density in the middle and lower density at the two ends. All other pairs of scatter plots have the similar pattern, so we can then assume MVN distribution. Mardia's MVN test can further reinforce this assumption:

```
mvt=mvn(st.residuals)
mvt$multivariateNormality

##              Test         Statistic           p value Result
## 1 Mardia Skewness  702.788193942711 0.999999999989264    YES
## 2 Mardia Kurtosis 0.860742082447038 0.389380110116907    YES
## 3             MVN              <NA>              <NA>    YES
```

Mardia's test performs test for skewness and kurtosis and combine them together to test multivariate normality. If both tests results state "YES", then the data follows a MVN distribution at significance level 0.05. Therefore, we can safely assume MVN distribution for the standardised residuals.

3. Maximum likelihood estimation of MVN distribution parameters

To find the parameter in the MVN model, we can assume $(\varepsilon_1^*, \ldots, \varepsilon_{17}^*) \sim MVN(0, \sigma)$ where σ is a 17×17 correlation matrix with $\sigma_{i,j} = exp\{-r\{dist_{i,j}\}^{\alpha}\}, where\ 0 < \alpha < 2, r > 0,$ and $dist_{i,j}$ is the distance between station I and j. Then the absolute value of log-likelihood function for the MVN density in R can be calculated as follow:

```
dist = matrix(NA, ncol = 17, nrow = 17)
for (i in 1:17){
  for (j in 1:17){
   dist[i,j] = distVincentyEllipsoid(c(elon[i],nlat[i]),c(elon[j],nlat[j]))/100000
  }
}

loglik = function(param){
  a = param[1]
  r = param[2]
  sigma1 = exp((-1)*r*dist^a)
  loglik = abs(sum(dmvnorm(st.residuals,mean = rep(0,17), sigma = sigma1, log = TR
UE)))
  return(loglik)
  }
```

Maximising the likelihood is equivalent to minimising negative log-likelihood, so we can minimise the above function to find MLE (maximum likelihood estimates) for α and r as follow:

```
start = c(0.5,1)
NM=optim(start,fn = loglik, method = "Nelder-Mead")
BFGS=optim(start,fn = loglik, method = "BFGS")
CG=optim(start,fn = loglik, method = "CG")
SANN=optim(start,fn = loglik, method = "SANN")
NLM=nlm(loglik,start)
```

Using $\sigma_{i,j} = exp\{-\sqrt{dist_{i,j}}\}, i.e. \alpha = 0.5, r = 1$, as initial estimates* for optim() in R gives:

```
results

##                            NM       BFGS            CG        SANN
## a               1.136600e+00 1.03191173 1.0372172317 1.852809727
## r               2.858057e-01 0.25150526 0.2531069769 0.782525529
## log-likelihood 7.915237e-06 0.00073821 0.0005638436 0.003883429
##                           NLM
## a               1.282132068
## r               0.344642054
## log-likelihood 0.002425739
```

*Note: different initial estimates give different estimates. This might be due to the local minimum in the bumpy data, so I have also tried different initial estimates (e.g. (1,1), (1,0.5), (0.5,0.5)) but the initial estimates (0.5,1) gives the smallest negative log-likelihood.

As we can see, Nelder-Mead method returns the smallest log-likelihood value. Hence, its estimates are chosen as MLEs and are used for following bootstraps estimation.

4. Bootstrap confidence intervals of MVN distribution parameters

To construct the bootstrap confidence interval for $\alpha \; and \; r$, residuals bootstrap is more appropriate because standardised residuals are assumed to be independent on time and station as we can see from the acf plots from question 1 and the MVN conclusions from question2. It would be inappropriate to resample the cases because the time spatial data apparently depends on each other. As we assume standardised residuals follow MNV distribution, we can estimate the distribution by $MVN(0, \hat{\sigma})$ instead of empirical distribution. Then the bootstrap steps are as follow:

1. Continue to use standardised residuals from model 3, $\varepsilon_{ij}^{*}, i = 1,..150, j = 1,...17.$
2. Generate parametric bootstrap samples from $MVN(0, \hat{\sigma})$, where $\hat{\sigma}_{i,j} = exp\{-\hat{r}\{dist_{i,j}\}^{\hat{\alpha}}\}$, $\hat{\alpha}$ and $\hat{r}$ are MLE by Nelder-Mead optimisation method. So $\varepsilon_{ij}^{b} \sim MVN(0, \hat{\sigma}), for \; i = 1,..150, j = 1,...17.$
3. Create a bootstrap sample of responses: $Y_{ij}^{*} = \left(\gamma_0 + \beta_1 lag1_j + \beta_2 lag2_j + \gamma_1 elevation_j\right) + \varepsilon_{ij}^{b} \times sd\left(\varepsilon_{ij}\right), for \; i = 1,..150, j = 1, \ldots 17.$
   Note: because ε_{ij}^{b} is standardised residuals, it wouldn't be appropriate to add directly in the model, so I scale it with standard deviation of original residuals from model3 to approximate.
4. Fit model3 to $\{Y_{ij}^{*}, lag1_j, lag2_j, elevation_j\}_i$, and standardise the residuals to obtain $\varepsilon_{ij}^{**}, for \; i = 1,..150, j = 1,...17.$
5. Minimise the log-likelihood of ε_{ij}^{**} to obtain the MLE of α^b, r^b.
6. Repeat this for 2000 times to get the distributions of $\alpha \; and \; r$.

Boot() in R can combine the above steps into one process as follow:

```
z = cbind(Y,lag1,lag2,Elev)
lm1.bt = function(x){
  temp = lm(x[,1]~x[,2]+x[,3]+x[,4])
  temp1=temp$residuals
```

```
  temp2=temp1/sd(temp1)
  loglik = function(param){
  a = param[1]
  r = param[2]
  sigma1 = exp(-r*dist^a)
  loglik = abs(sum(dmvnorm(matrix(temp2,ncol = 17), mean = rep(0,17), sigma = sigm
a1, log = TRUE)))
  }
  NM=optim(start,fn = loglik, method = "Nelder-Mead")
  return(NM$par)
}

lm1.gen = function(x,mle){
  a = mle[1]
  r = mle[2]
  sigma1 = exp(-r*dist^a)
  n = 150
  err = rmvnorm(n, mean = rep(0,17), sigma = sigma1)
  err = as.vector(err)
  y.star = mle[3]+mle[4]*x[,2]+mle[5]*x[,3]+mle[6]*x[,4]+err*sd(lm.fit1$residuals)

  return(cbind(y.star,x[,2],x[,3],x[,4]))

}

mle.list = c(optim(start,fn = loglik, method = "Nelder-Mead")$par,coef(lm.fit1))
set.seed(1234)
boot1 = boot(data=z,statistic=lm1.bt,R=2000,sim="parametric",ran.gen=lm1.gen,mle=m
le.list)
boot1

##
## PARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = z, statistic = lm1.bt, R = 2000, sim = "parametric",
##     ran.gen = lm1.gen, mle = mle.list)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 1.1271220 0.009413047  0.03054340
## t2* 0.2808098 0.011921844  0.03043129
```

As we can see bias and standard error are quite small, which suggests that MLEs are good estimators.

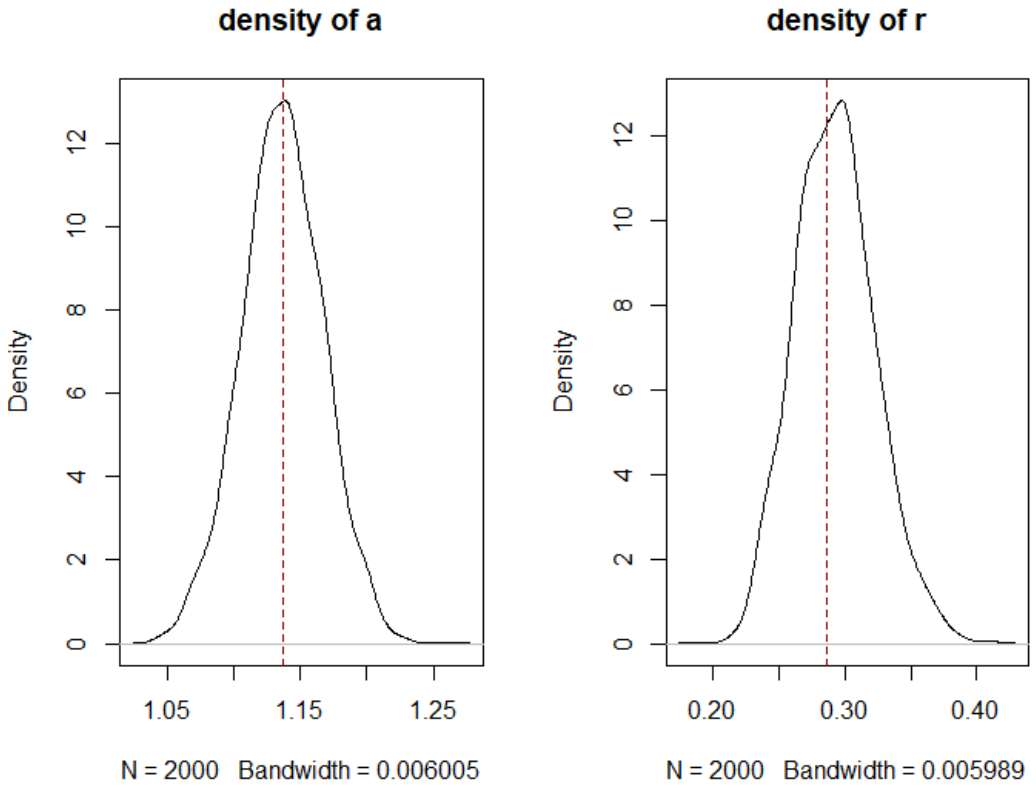
Bootstrap confidence interval for α is:

```
## Intervals :
## Level      Normal              Basic              Percentile
## 95%   ( 1.058,  1.178 )   ( 1.057,  1.180 )   ( 1.074,  1.197 )
## Calculations and Intervals on Original Scale
```

Bootstrap confidence interval for r is:

```
## Intervals :
## Level      Normal              Basic              Percentile
## 95%   ( 0.2092,  0.3285 )   ( 0.2043,  0.3245 )   ( 0.2372,  0.3573 )
## Calculations and Intervals on Original Scale
```

The density plots of $\alpha \; and \; r$ are as follow, the red dotted line is the MLE:



It suggests that $\alpha \; and \; r$ are likely to follow normal distribution, and their MLEs are around the medium.

5. Distance dependency presented in correlation matrix

Finally, we will use the generated standard residuals to compose a correlation matrix for all stations.

(TBC)