코로나 확진자수 예측

자료읽기

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## √ ggplot2 3.3.2     √ purrr   0.3.4
## √ tibble  3.0.3     √ dplyr   1.0.2
## √ tidyr   1.1.2     √ stringr 1.4.0
## √ readr   1.3.1     √ forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(fpp3)
```

```
## Warning: package 'fpp3' was built under R version 4.0.3
```

```
## -- Attaching packages ---------------------------------------------- fpp3 0.3 --
```

```
## √ lubridate   1.7.9     √ feasts      0.1.5
## √ tsibble     0.9.2     √ fable       0.2.1
## √ tsibbledata 0.2.0
```

```
## Warning: package 'fable' was built under R version 4.0.3
```

```
## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x lubridate::date()   masks base::date()
## x dplyr::filter()     masks stats::filter()
## x tsibble::interval() masks lubridate::interval()
## x dplyr::lag()        masks stats::lag()
```

```
df <- read_csv("kr_daily.csv")
```

```
## Parsed with column specification:
## cols(
##   date = col_double(),
##   confirmed = col_double(),
##   death = col_double(),
##   released = col_double(),
##   tested = col_double(),
##   negative = col_double()
## )
```

```
df <- df[,1:2]
```

```
df
```

```
## # A tibble: 326 x 2
##        date confirmed
##       <dbl>     <dbl>
##  1 20200121         1
##  2 20200122         1
##  3 20200123         1
##  4 20200124         2
##  5 20200125         2
##  6 20200126         2
##  7 20200127         4
##  8 20200128         4
##  9 20200129         4
## 10 20200130         7
## # ... with 316 more rows
```

```
TSB <- mutate(df, date=ymd(date)) %>%
  as_tsibble(index=date)
```

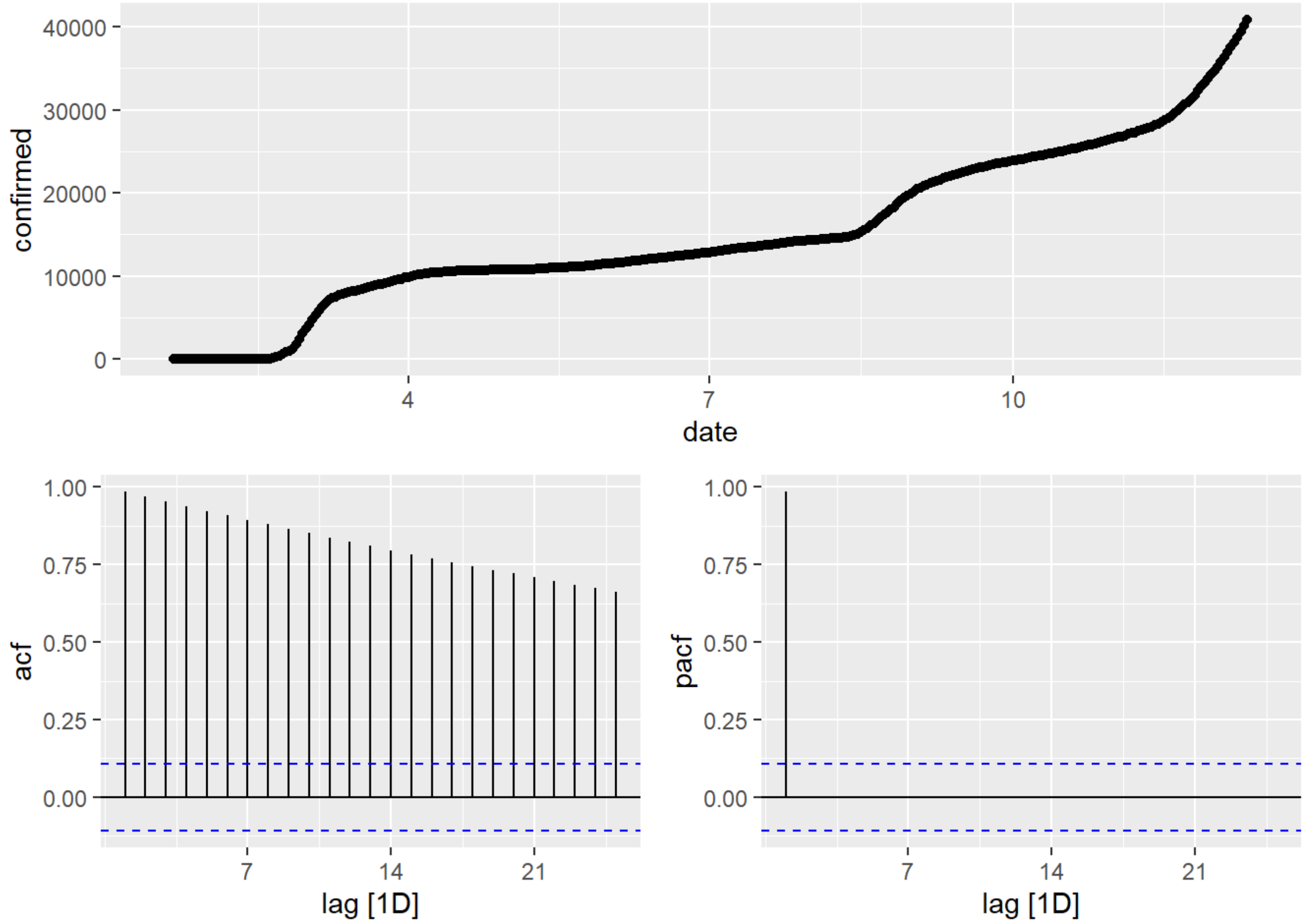
```
TSB
```

```
## # A tsibble: 326 x 2 [1D]
##    date       confirmed
##    <date>         <dbl>
##  1 2020-01-21         1
##  2 2020-01-22         1
##  3 2020-01-23         1
##  4 2020-01-24         2
##  5 2020-01-25         2
##  6 2020-01-26         2
##  7 2020-01-27         4
##  8 2020-01-28         4
##  9 2020-01-29         4
## 10 2020-01-30         7
## # ... with 316 more rows
```

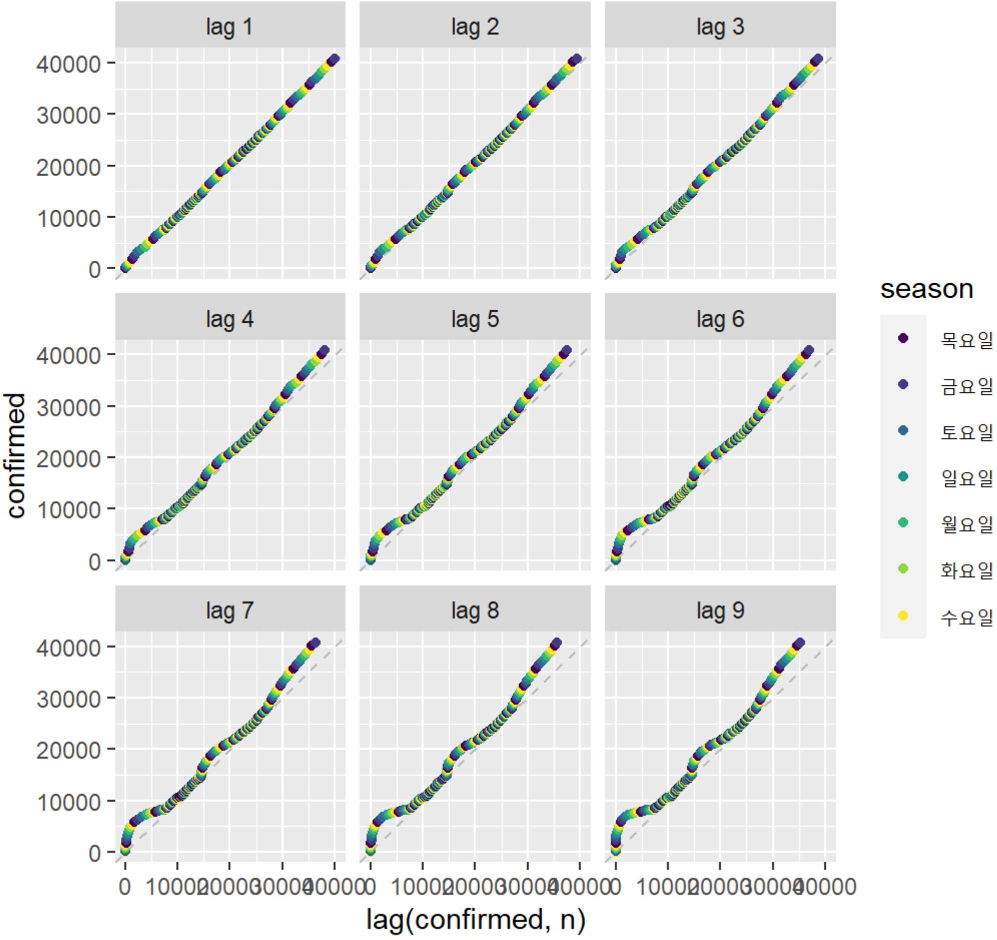
탐색/분할

- 시계열 시각화

```
gg_tsdisplay(TSB, confirmed, plot_type='partial')
```



```
gg_lag(TSB, confirmed, geom='point')
```



```
# 분산안정화
#lambda <- features(.tbl=TSB, .var=confirmed, features=guerrero) %>%
#  pull(lambda_guerrero)

# autoplot(TSB, box_cox(confirmed, lambda))
```

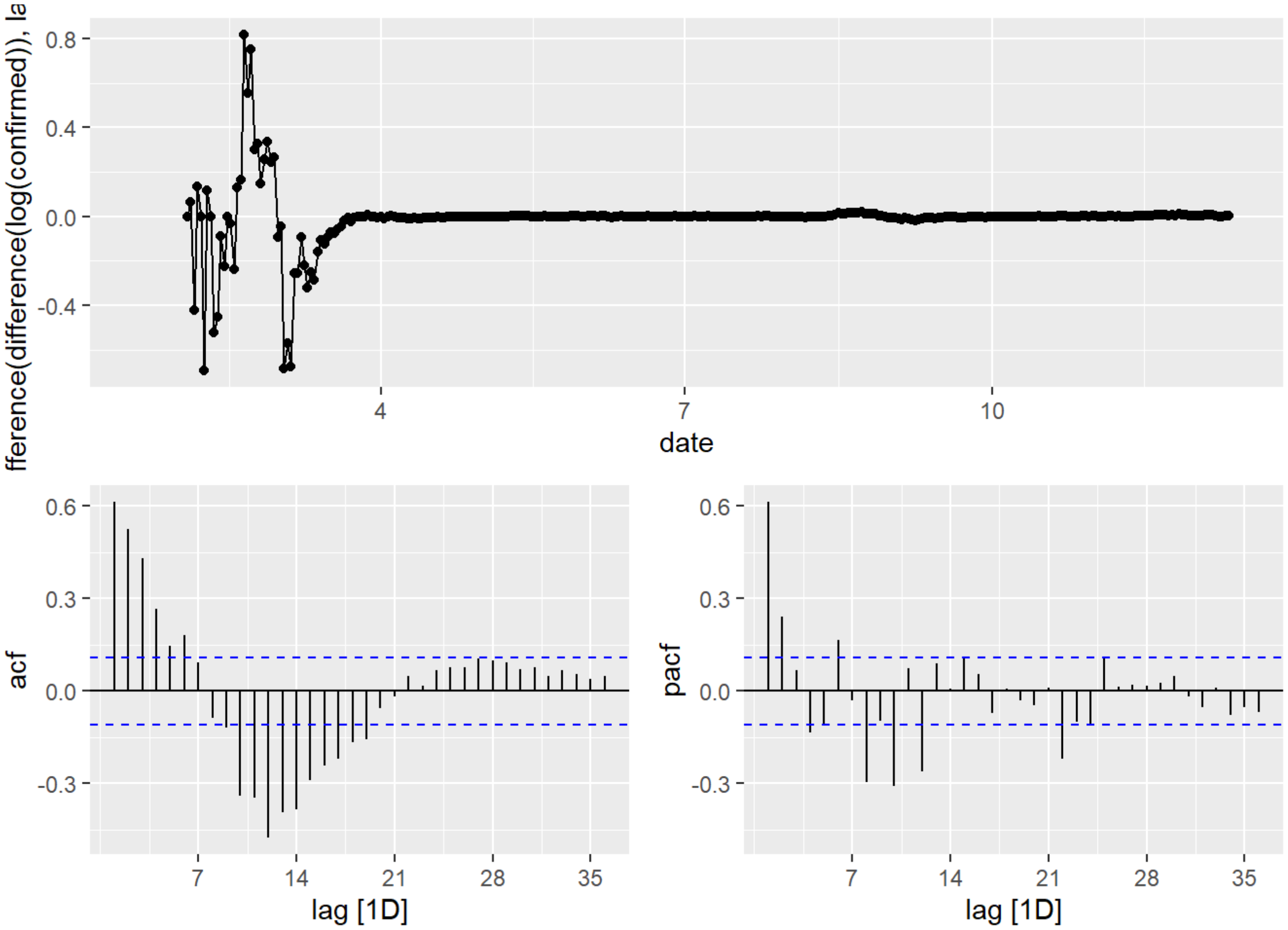
```
# 차분차수 d 결정
features(TSB, log(confirmed), unitroot_ndiffs)
```

```
## # A tibble: 1 x 1
##   ndiffs
##    <int>
## 1      2
```

```
gg_tsdisplay(TSB, difference(difference(log(confirmed)),lag=12), plot_type='partial', lag=36)
```

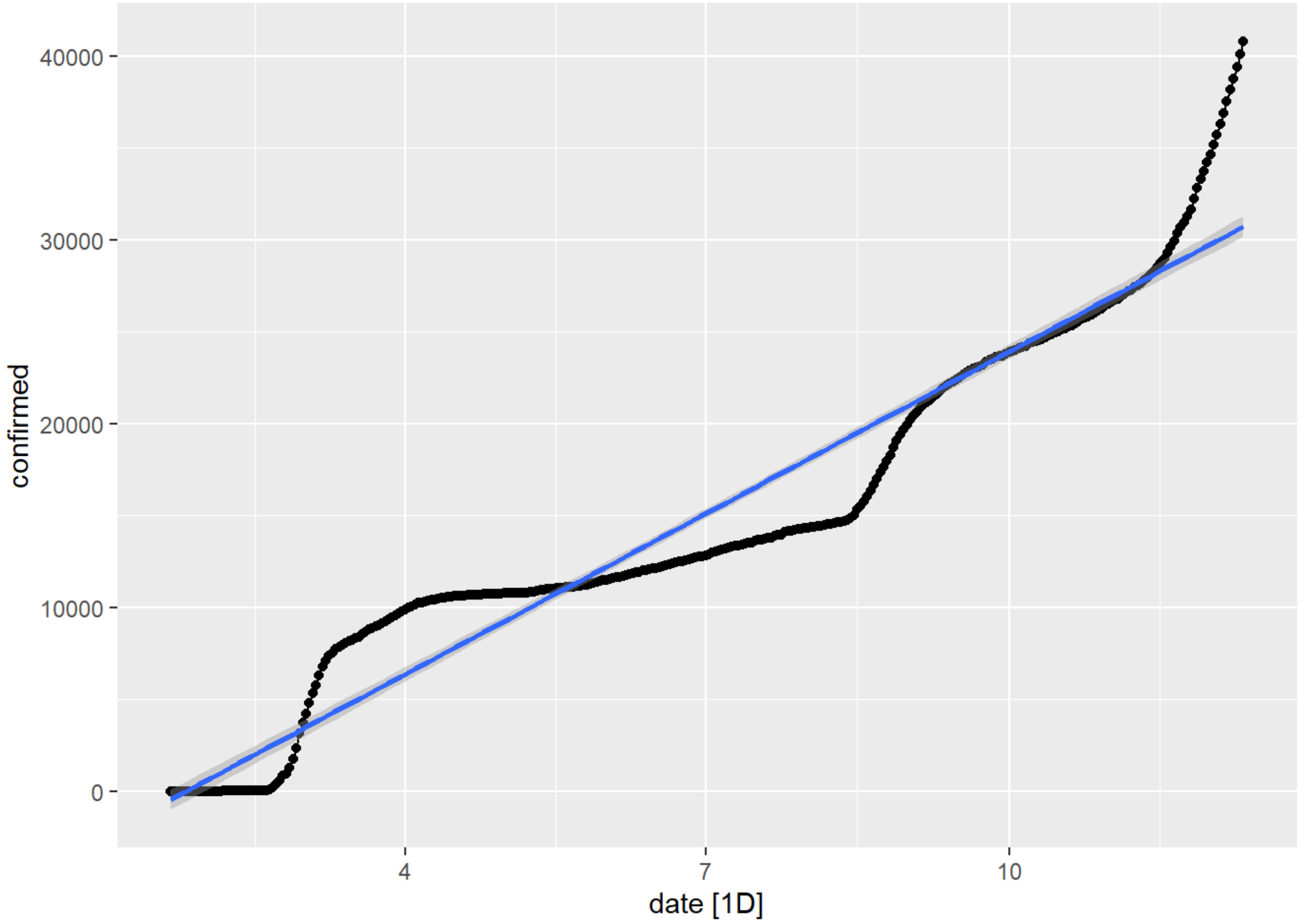
```
## Warning: Removed 13 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 13 rows containing missing values (geom_point).
```



```
TRN <- filter_index(TSB, .~'2020-11-30')
TST <- filter_index(TSB, '2020-12-01'~'2020-12-10')

autoplot(TSB, confirmed) + geom_point() + geom_smooth(method='lm')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



모형

모형 적합

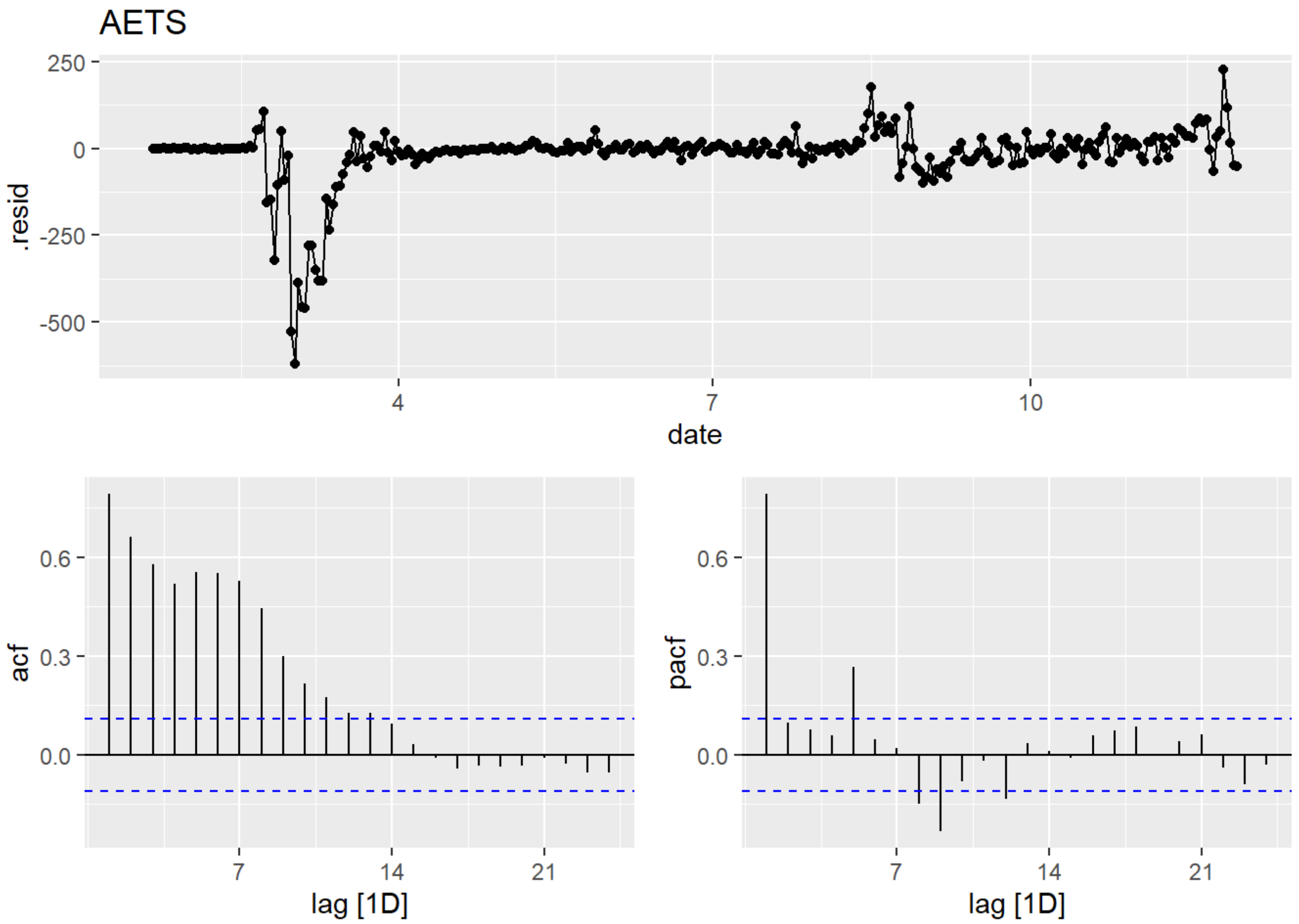
```
M <- model(TRN,
           AETS = ETS(log(confirmed)),
           ADN = ETS(log(confirmed)~error('A')+trend('Ad', phi=0.9)+season('N')),
           AARIMA = ARIMA(log(confirmed)),
           M121000 = ARIMA(log(confirmed) ~ pdq(1,2,1)+PDQ(0,0,0)),
           NSARIMA = ARIMA(log(confirmed) ~ PDQ(0,0,0)))
```

모형 탐색

```
A <- augment(M)
```

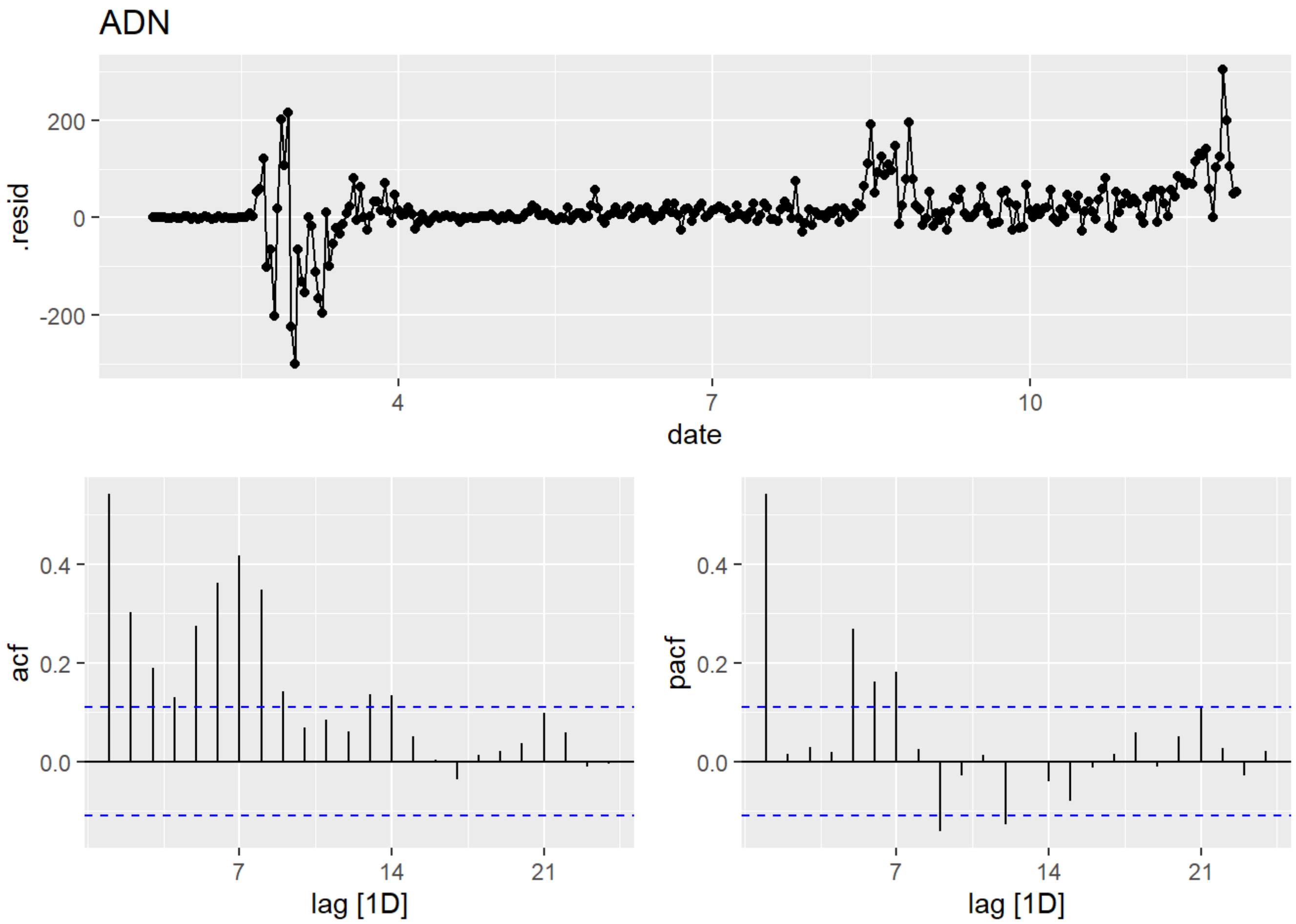
- ETS(자동선택) 잔차분석

```
gg_tsdisplay(filter(A, .model=='AETS'), .resid, plot_type='partial') + ggtitle('AETS')
```



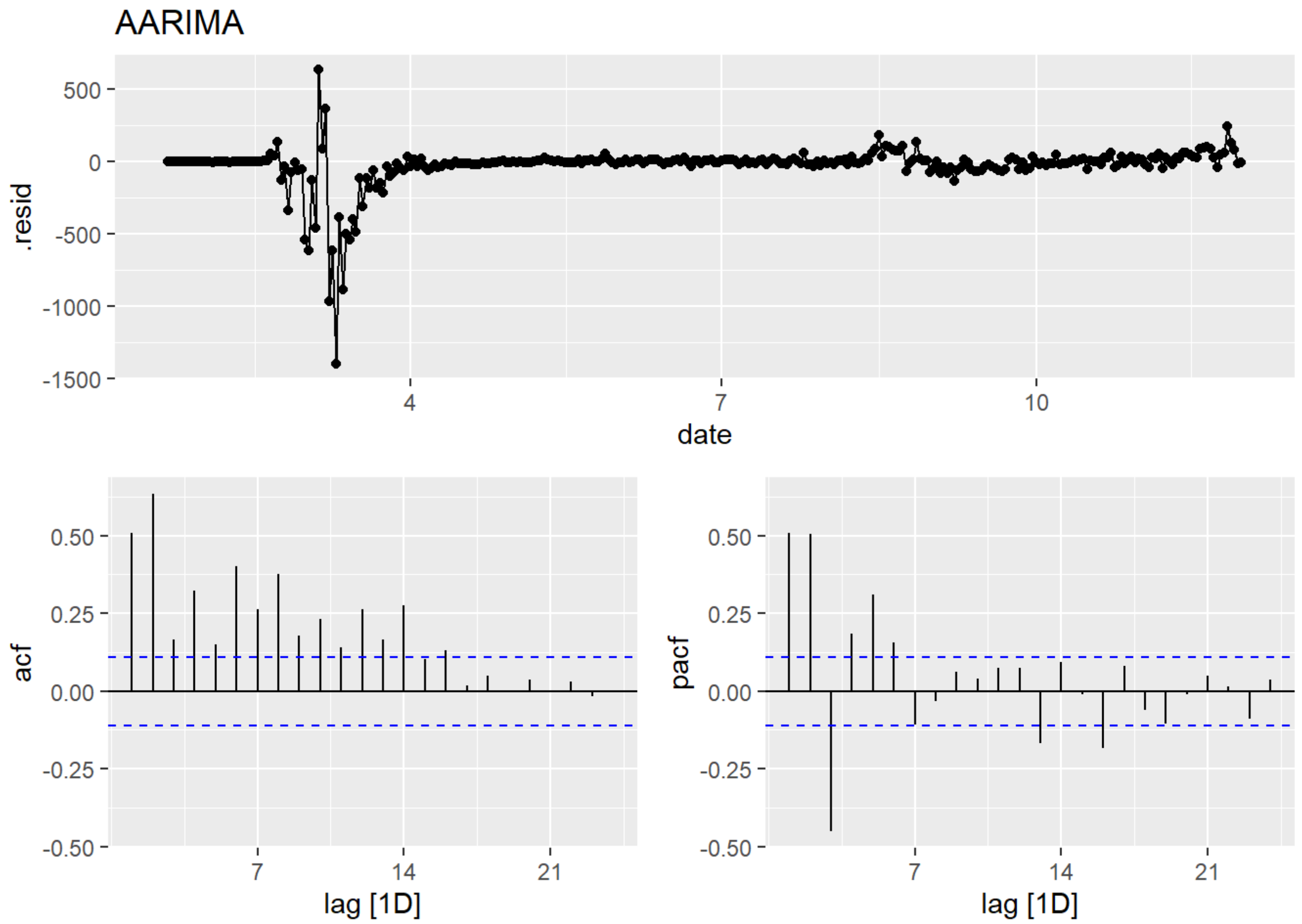
- ETS(A,Ad,N) 잔차분석

```
gg_tsdisplay(filter(A, .model=='ADN'), .resid, plot_type='partial') + ggtitle('ADN')
```



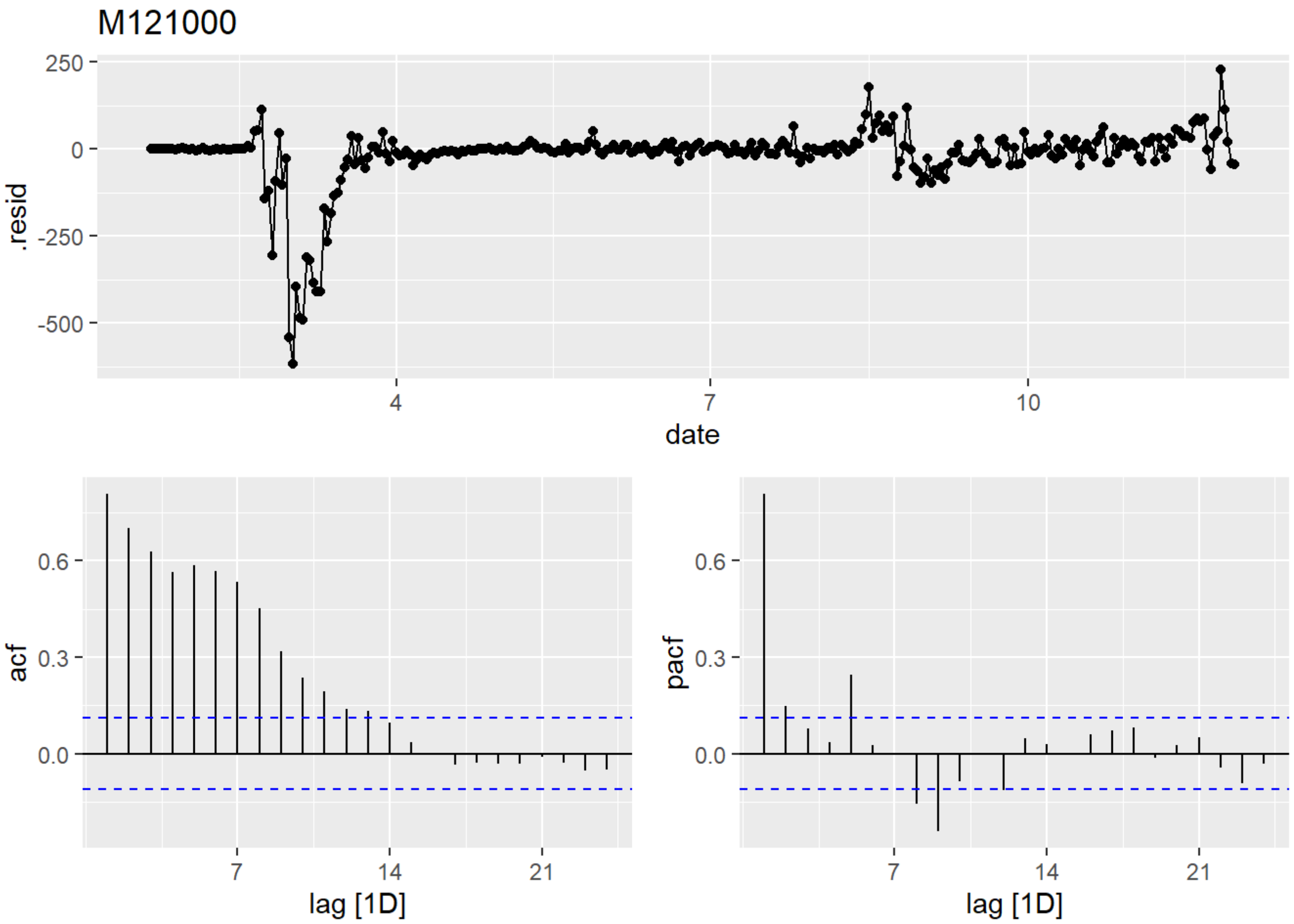
- ARIMA(자동선택) 잔차분석

```
gg_tsdisplay(filter(A, .model=='AARIMA'), .resid, plot_type='partial') + ggtitle('AARIMA')
```



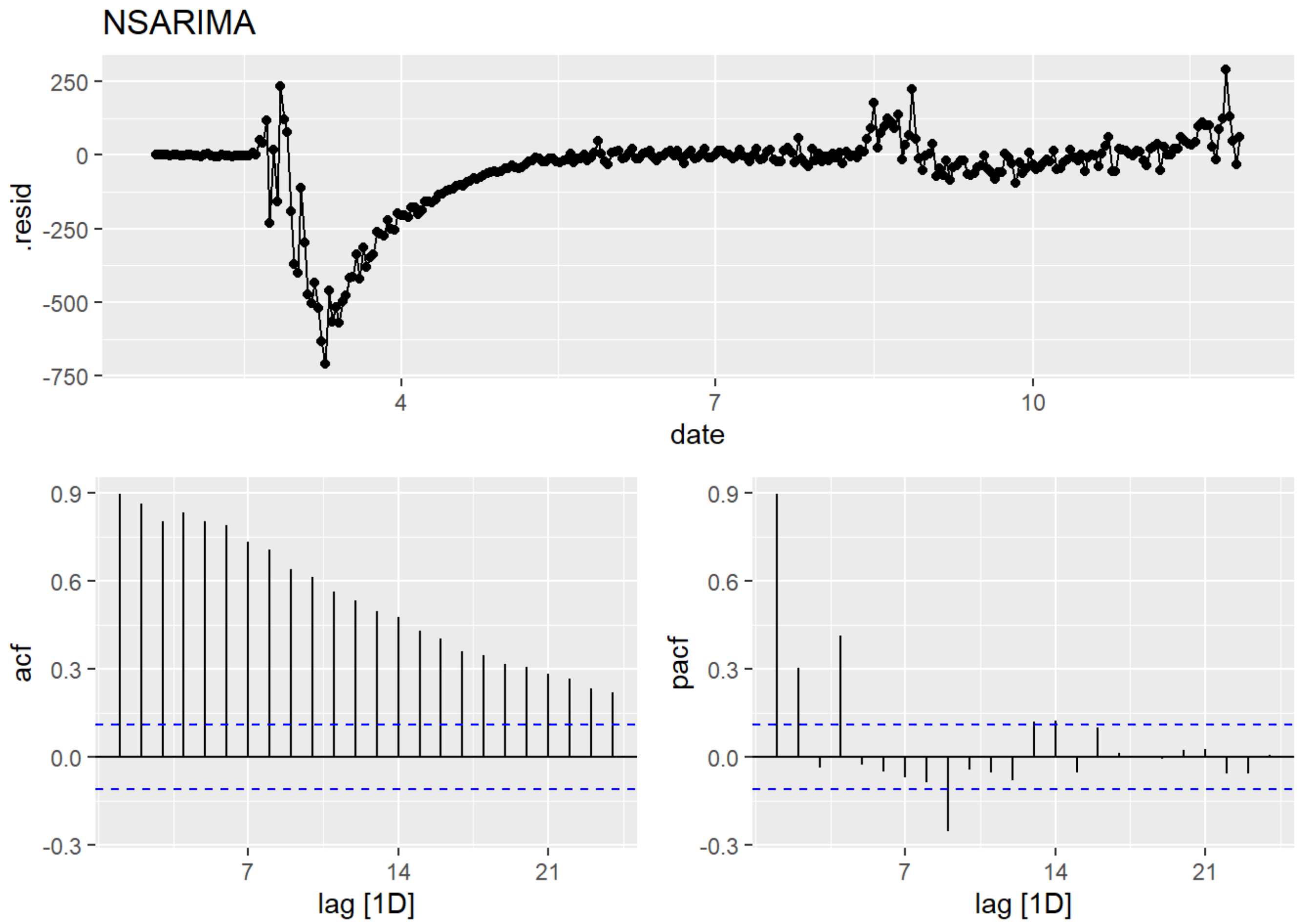
- ARIMA(1,2,1)(0,0,0) 잔차분석

```
gg_tsdisplay(filter(A, .model=='M121000'), .resid, plot_type='partial') + ggtitle('M121000')
```



- ARIMA(pdq 자동선택)(0,0,0) 잔차분석

```
gg_tsdisplay(filter(A, .model=='NSARIMA'), .resid, plot_type='partial') + ggtitle('NSARIMA')
```



최종모형 결정

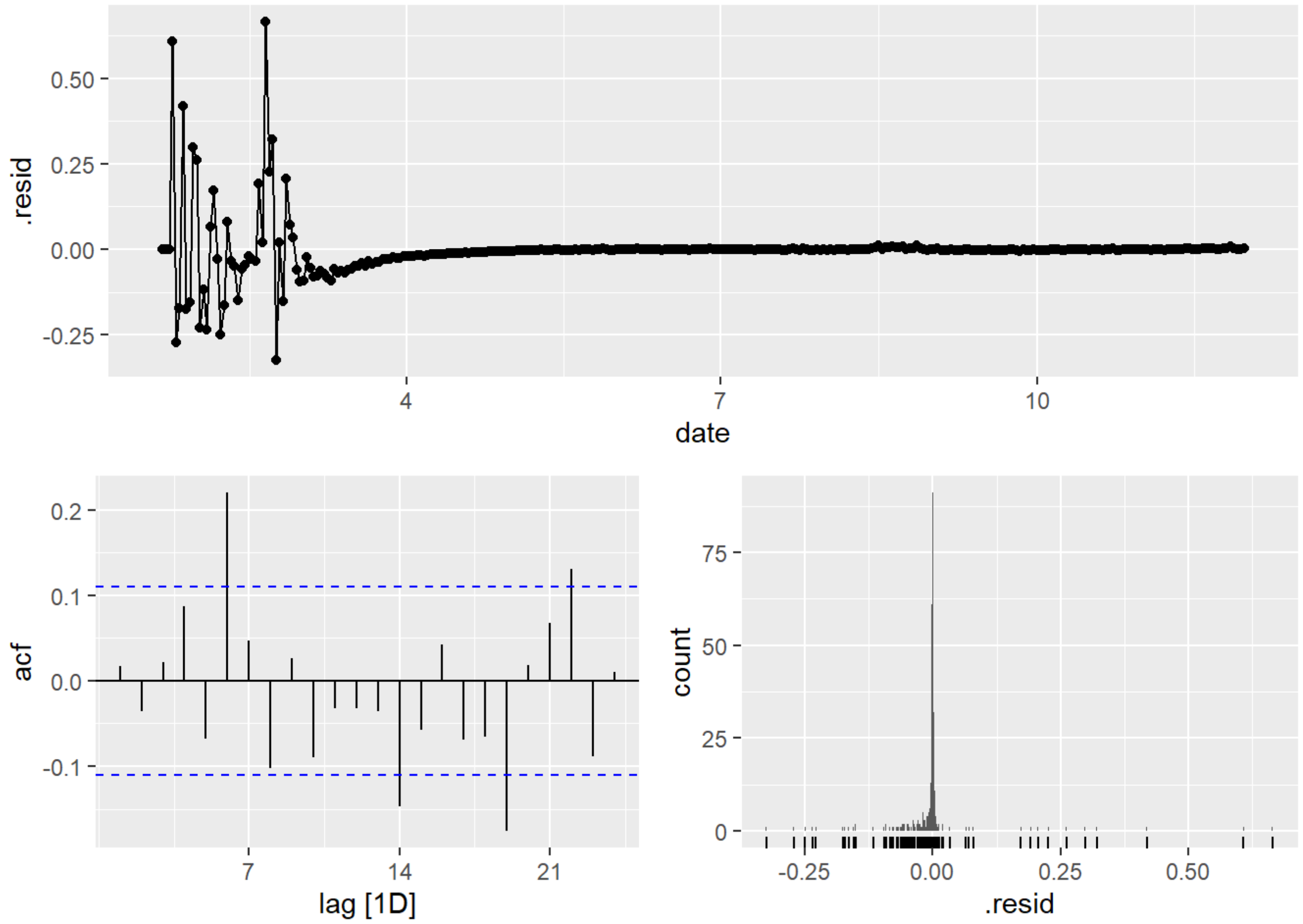
```
glance(M)
```

```
## # A tibble: 5 x 11
##   .model  sigma2 log_lik   AIC  AICc   BIC      MSE    AMSE     MAE ar_roots
##   <chr>    <dbl>   <dbl> <dbl> <dbl> <dbl>    <dbl>   <dbl>   <dbl> <list>
## 1 AETS   0.00765   -136.  283.  283.  302.  0.00755  0.0234  0.0255 <NULL>
## 2 ADN    0.00742   -131.  273.  273.  291.  0.00731  0.0214  0.0227 <NULL>
## 3 AARIMA 0.00710    332. -650. -650. -624. NA       NA      NA      <cpl [1~
## 4 M1210~ 0.00764    320. -633. -633. -622. NA       NA      NA      <cpl [1~
## 5 NSARI~ 0.00667    341. -672. -672. -654. NA       NA      NA      <cpl [0~
## # ... with 1 more variable: ma_roots <list>
```

ETS 모형 중에 AICc의 값이 가장 작은 모형은 additive damped trend 모형인 ADN이 최종모형이고,
ARIMA 모형 중에 AICc의 값이 가장 작은 모형은 시계열을 고려하지 않고 pdq만 자동으로 선택한 모형인 NSARIMA가 최종모형이다.
전체 모형으로 따졌을 땐 가장 AICc가 작은 NSARIMA 모형이다.

ARIMA 모형 중 최종모형 잔차분석 (NSARIMA)

```
# ARIMA 모형의 잔차의 ACF
ARIMA <- select(M, NSARIMA)
gg_tsresiduals(ARIMA)
```



```
# ARIMA 모형의 잔차의 Ljung-Box 검정
A_ARIMA <- augment(ARIMA)
features(A_ARIMA, .resid, ljung_box, lag=24, dof=4)
```

```
## # A tibble: 1 x 3
##   .model  lb_stat lb_pvalue
##   <chr>     <dbl>     <dbl>
## 1 NSARIMA   2637.         0
```

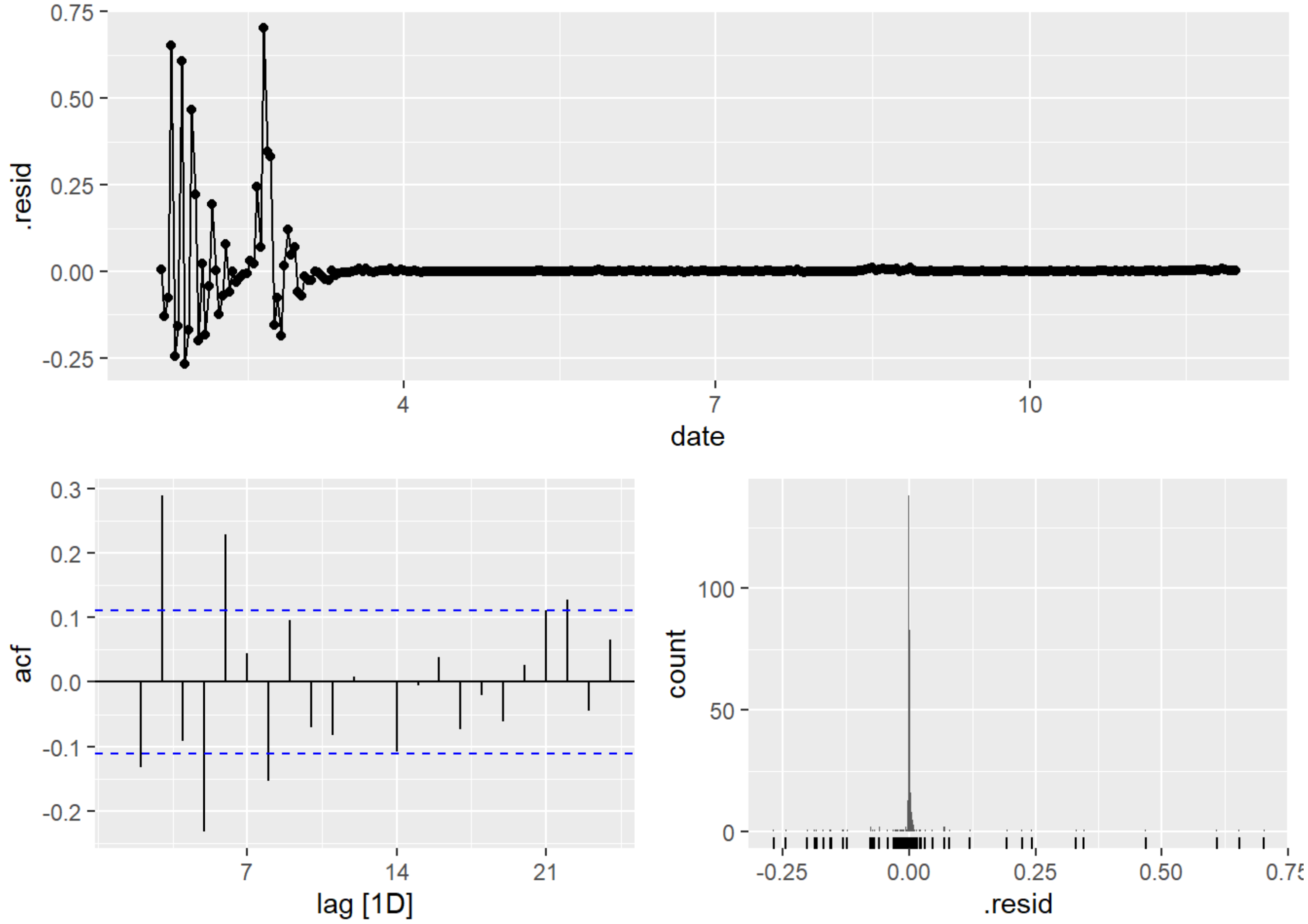
백색잡음이 아니다. (남은 정보가 있다)

```
report(ARIMA)
```

```
## Series: confirmed
## Model: ARIMA(0,2,4)
## Transformation: log(.x)
##
## Coefficients:
##           ma1      ma2     ma3      ma4
##       -0.6318  -0.0403  0.1951  -0.4040
## s.e.   0.0505   0.0635  0.0553   0.0545
##
## sigma^2 estimated as 0.006672:  log likelihood=341.12
## AIC=-672.23   AICc=-672.04   BIC=-653.5
```

ETS 모형 중 최종모형 잔차분석 (ADN)

```
# ETS 모형의 잔차의 ACF
ETS <- select(M, ADN)
gg_tsresiduals(ETS)
```



```
# 잔차의 Ljung-Box 검정
A_ETS <- augment(ETS)
features(A_ETS, .resid, ljung_box, lag=24, dof=3)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 ADN       332.         0
```

백색잡음이 아니다. (남은 정보가 있다)

```
report(ETS)
```

```
## Series: confirmed
## Model: ETS(A,Ad,N)
## Transformation: log(.x)
##   Smoothing parameters:
##     alpha = 0.9749094
##     beta  = 0.383609
##     phi   = 0.9
##
##   Initial states:
##           l         b
##  -0.1477603 0.1581436
##
##   sigma^2:  0.0074
##
##      AIC     AICc      BIC
## 272.5571 272.7513 291.3199
```

예측

- 예측값 저장(TST)/모형 평가

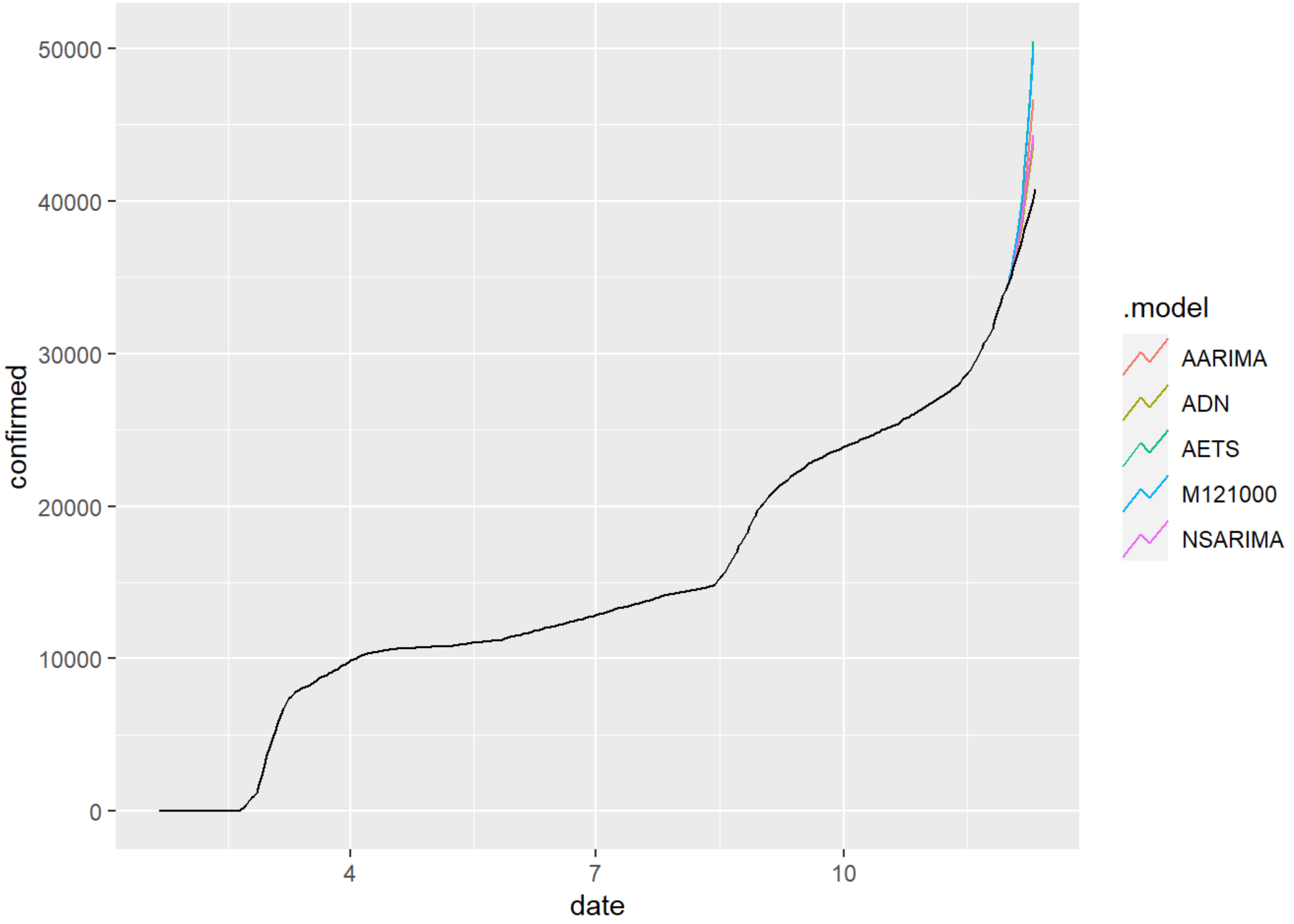
```
FF <- forecast(M, new_data=TST)
accuracy(FF, data=TSB)
```

```
## # A tibble: 5 x 9
##   .model  .type     ME  RMSE   MAE   MPE  MAPE  MASE  ACF1
##   <chr>   <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 AARIMA  Test  -2317. 3121. 2317. -5.98  5.98  3.11 0.668
## 2 ADN     Test  -1267. 1748. 1267. -3.26  3.26  1.70 0.678
## 3 AETS    Test  -3537. 4833. 3537. -9.12  9.12  4.75 0.658
## 4 M121000 Test  -3398. 4615. 3398. -8.77  8.77  4.56 0.660
## 5 NSARIMA Test  -1558. 2098. 1558. -4.02  4.02  2.09 0.697
```

AICc값으로 최종모형을 결정하였지만 12-01~12-10 에 대한 예측값의 MAPE값을 확인해보았더니 ADN이 3.26로 가장 작고, NSARIMA는 4.01로 그 다음으로 작았다.

- 예측값 시각화

```
# 한 번에 모든 모형 시각화
autoplot(FF, data=TSB, level=NULL)
```



- ARIMA 최종모형(NSARIMA)와 ETS 최종모형(ADN) 의 예측값 시각화

```
# NSARIMA 예측
FARIMA <- forecast(ARIMA, new_data=TST)
as.data.frame(FARIMA)
```

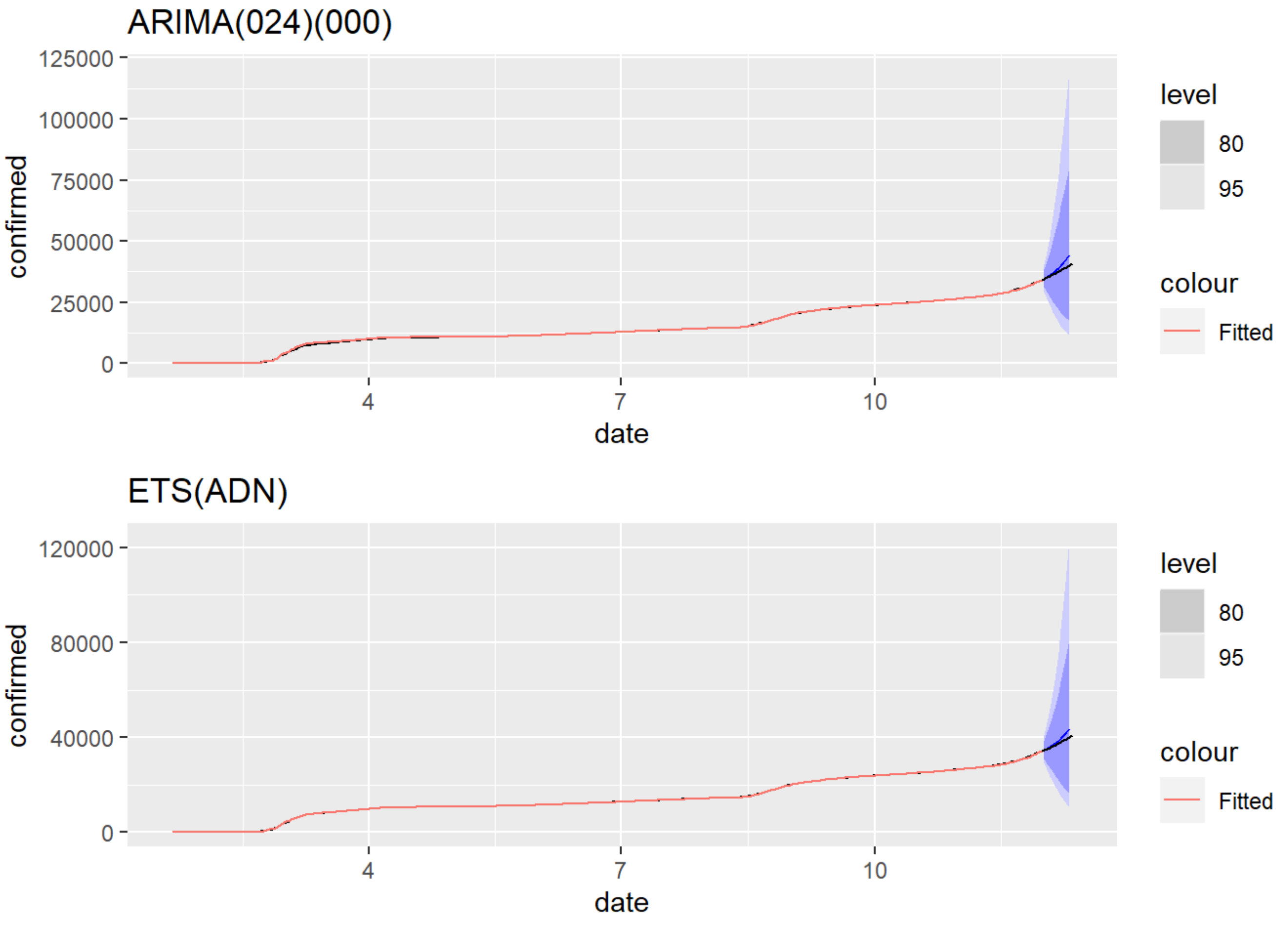
```
##     .model       date        confirmed    .mean
## 1  NSARIMA 2020-12-01 t(N(10, 0.0067)) 34675.07
## 2  NSARIMA 2020-12-02  t(N(10, 0.019)) 35227.44
## 3  NSARIMA 2020-12-03  t(N(10, 0.038)) 35931.46
## 4  NSARIMA 2020-12-04  t(N(10, 0.071)) 36862.46
## 5  NSARIMA 2020-12-05   t(N(10, 0.11)) 37873.06
## 6  NSARIMA 2020-12-06   t(N(10, 0.15)) 38968.76
## 7  NSARIMA 2020-12-07   t(N(11, 0.19)) 40155.23
## 8  NSARIMA 2020-12-08   t(N(11, 0.24)) 41438.31
## 9  NSARIMA 2020-12-09   t(N(11, 0.29)) 42824.02
## 10 NSARIMA 2020-12-10   t(N(11, 0.35)) 44318.58
```

```
G1 <- autoplot(FARIMA, data=TSB) + geom_line(aes(y=.fitted, color='Fitted'), data=A_ARIMA) + gg
title('ARIMA(024)(000)')
```

```
# ADN 예측
FETS <- forecast(ETS, new_data=TST)
as.data.frame(FETS)
```

```
##    .model       date        confirmed    .mean
## 1     ADN 2020-12-01 t(N(10, 0.0074)) 34697.95
## 2     ADN 2020-12-02   t(N(10, 0.02)) 35261.43
## 3     ADN 2020-12-03   t(N(10, 0.04)) 35917.74
## 4     ADN 2020-12-04  t(N(10, 0.067)) 36681.44
## 5     ADN 2020-12-05    t(N(10, 0.1)) 37561.55
## 6     ADN 2020-12-06   t(N(10, 0.14)) 38562.79
## 7     ADN 2020-12-07   t(N(10, 0.19)) 39686.49
## 8     ADN 2020-12-08   t(N(11, 0.25)) 40931.43
## 9     ADN 2020-12-09   t(N(11, 0.32)) 42294.47
## 10    ADN 2020-12-10   t(N(11, 0.39)) 43771.10
```

```
G2 <- autoplot(FETS, data=TSB) + geom_line(aes(y=.fitted, color='Fitted'), data=A_ETS) + ggtitl
e('ETS(ADN)')

gridExtra::grid.arrange(G1,G2)
```



- 각 최종모형의 예측값 확인

```
cbind(
  TST[,c('date', 'confirmed')],
  'ETS(ADN)'=filter(FF, .model=='ADN')$.mean,
  'ARIMA(024000)'=filter(FF, .model=='NSARIMA')$.mean
)
```

```
##          date confirmed ETS(ADN) ARIMA(024000)
## 1  2020-12-01     34652 34697.95      34675.07
## 2  2020-12-02     35163 35261.43      35227.44
## 3  2020-12-03     35696 35917.74      35931.46
## 4  2020-12-04     36325 36681.44      36862.46
## 5  2020-12-05     36908 37561.55      37873.06
## 6  2020-12-06     37539 38562.79      38968.76
## 7  2020-12-07     38154 39686.49      40155.23
## 8  2020-12-08     38746 40931.43      41438.31
## 9  2020-12-09     39417 42294.47      42824.02
## 10 2020-12-10     40097 43771.10      44318.58
```

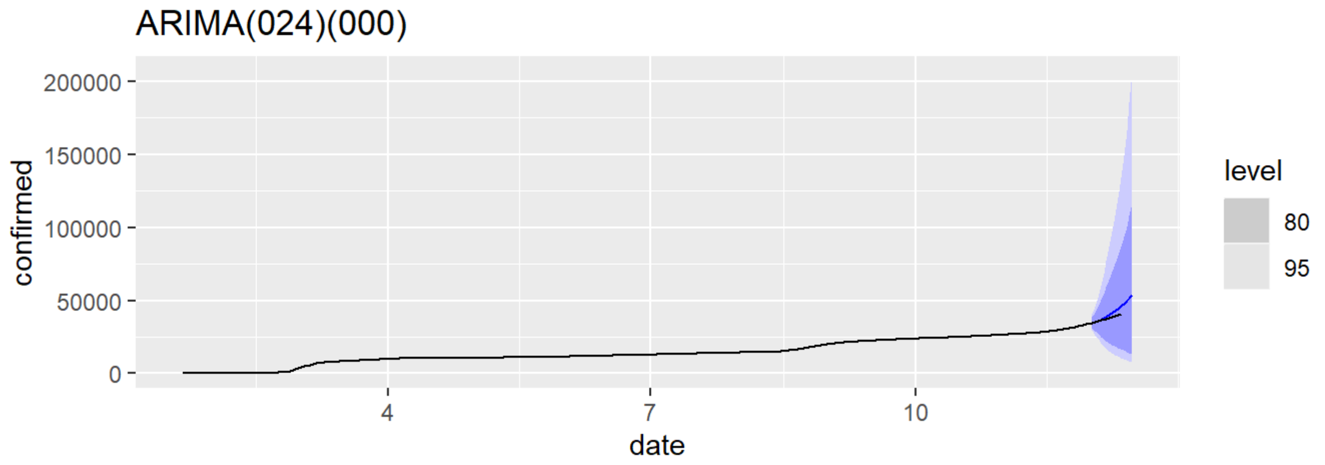
2020.12.1~12.15까지 확진자수 예측값과 예측그림

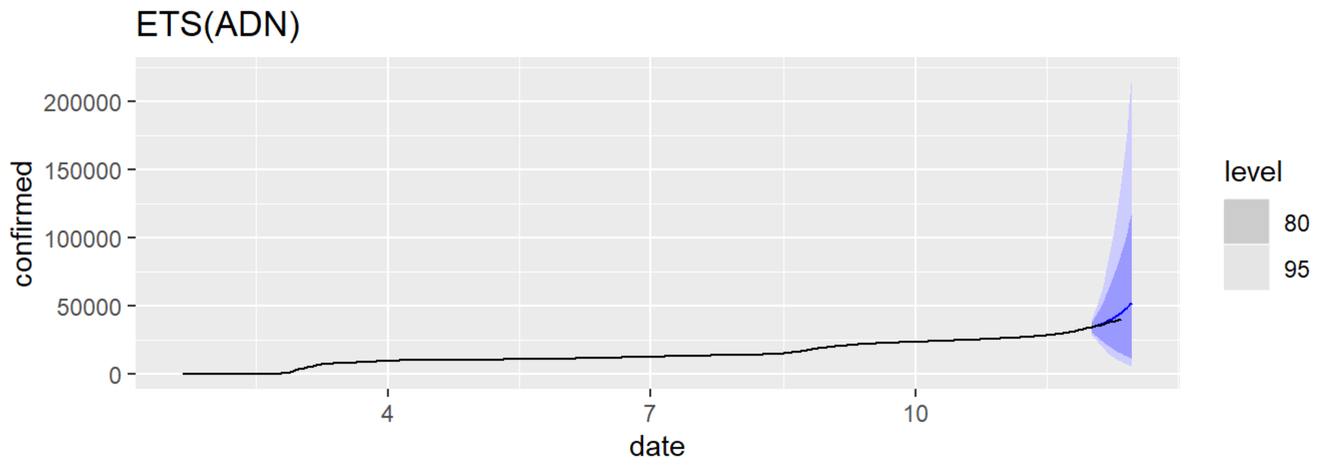
```
FA <- forecast(ARIMA, h=15)

P1 <- autoplot(FA, data=TSB) + ggtitle('ARIMA(024)(000)')
```

```
FE <- forecast(ETS, h=15)

P2 <- autoplot(FE, data=TSB) + ggtitle('ETS(ADN)')
```

- 예측 그림

```
gridExtra::grid.arrange(P1,P2)
```





- 예측값

```
cbind(
  FA[,c('date')],
  'ETS(ADN)' = FA$.mean,
  'ARIMA(024000)'=FE$.mean
)
```

```
##          date ETS(ADN) ARIMA(024000)
## 1  2020-12-01 34675.07      34697.95
## 2  2020-12-02 35227.44      35261.43
## 3  2020-12-03 35931.46      35917.74
## 4  2020-12-04 36862.46      36681.44
## 5  2020-12-05 37873.06      37561.55
## 6  2020-12-06 38968.76      38562.79
## 7  2020-12-07 40155.23      39686.49
## 8  2020-12-08 41438.31      40931.43
## 9  2020-12-09 42824.02      42294.47
## 10 2020-12-10 44318.58      43771.10
## 11 2020-12-11 45928.36      45355.88
## 12 2020-12-12 47659.96      47042.77
## 13 2020-12-13 49520.15      48825.38
## 14 2020-12-14 51515.91      50697.20
## 15 2020-12-15 53654.41      52651.74
```