Lancaster CMAF Friday Forecasting Talks

Experience in Applying Forecasting Techniques to Healthcare Data

Andy McCann
Lead Data Scientist | MLCSU Clinical

andrew.mccann1@nhs.net

17 March 2023

Initial Model-2019

Requirements and implementation

- Develop forecasting method applicable to all daily series, initial focus on A&E attendances and admissions
- Beat existing methods, remove arbitrary adjustments, extend time horizon
- Implemented in R, which can be integrated with SQL data warehouse
- Captures variation due to day of week, Bank Holiday weekends, school holidays, start of term, seasonal variation and any underlying trend
- ARIMA structure captures step-changes and short-term dynamics and calculates robust confidence intervals
- Model structure able to capture effects of other factors in future, e.g. weather and other repeating events (e.g. football derbies)

Caveats/Concerns

- Forecasts are always wrong!
- Lots of variability in the series
- Danger of ‘spurious precision’
- How are the forecasts used in practice? e.g. capacity constraints
- Volume of attendances not actually what usually causes A&E problems
- Transparency and ‘explainability’ important. Understanding and explaining underlying seasonality (day of week, time of year etc.) probably more important than actual forecasts.
- Crucial to evaluate based on forecasts, not on fit, and to compare forecasting performance with existing methods. Initial models were generally fit using data from 2010-2017 and forecast the whole of 2018.

Use of R

- I was a newcomer to R (and to forecasting)
- R obvious choice to use modern forecasting methods. Actual fitting model only one line of code:

```
DHRTest <- auto.arima(attendancests, seasonal=FALSE, lambda=0,
        xreg=as.matrix(cbind(UseDummy[1:lastactual,],fourier(attendancests, K=KTerms))))
```

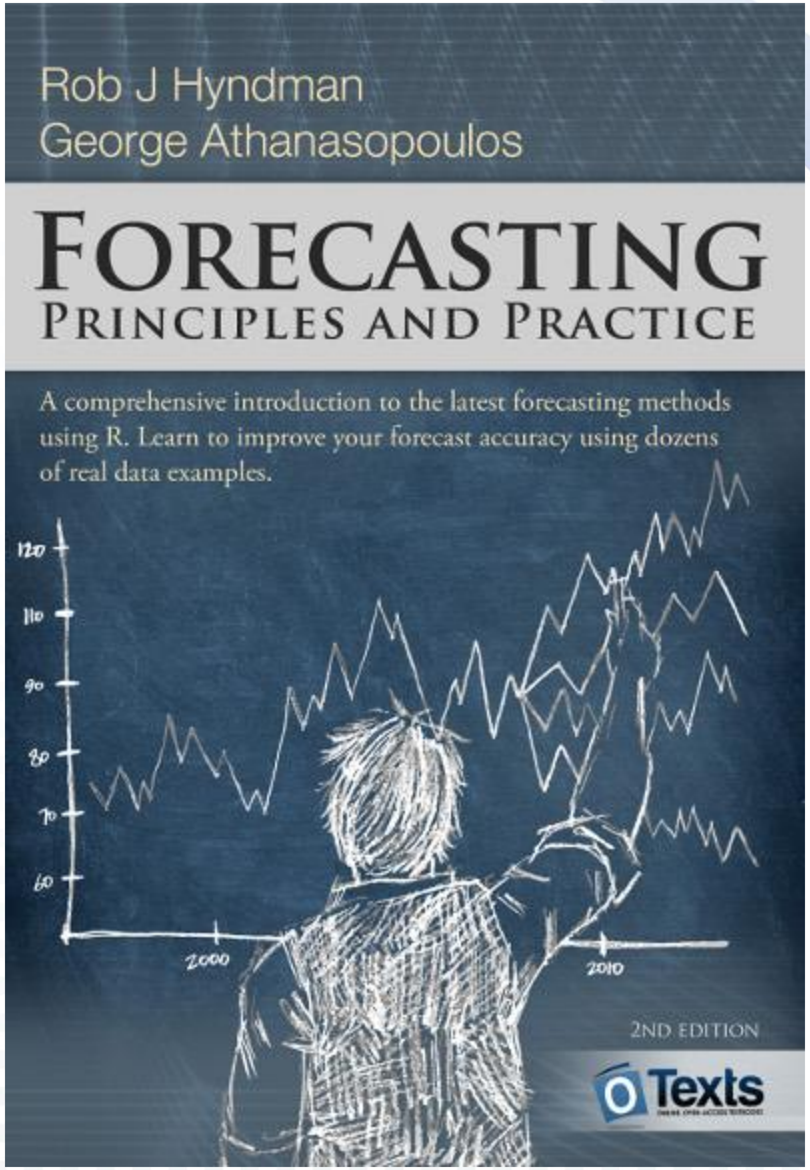
- Most code is data ‘wrangling’, creating dummy variables etc.
- Also found R (and markdown notebook) useful for charting/visualisations for data discovery (but still used Excel at times!)
- R particularly good for batch procedures to evaluate across multiple datasets
- I am used to coding, but still challenges around terminology, version changes, etc.

Intodoctory guide:

My guide was the second edition of this free online book, an introduction to forecasting plus a practical guide to the use of the R forecast package, by the person who created the package. Regularly updated.

There is now a third edition, which uses the fable package rather than forecast and integrates better with the tidyverse:

https://otexts.com/fpp3/



Methods

- Existing methodology was average of same weekday of preceding 6 weeks, sometimes manually adjusted, e.g. for Bank Holidays. Actually performs relatively well, though struggles for e.g. start of school holidays.

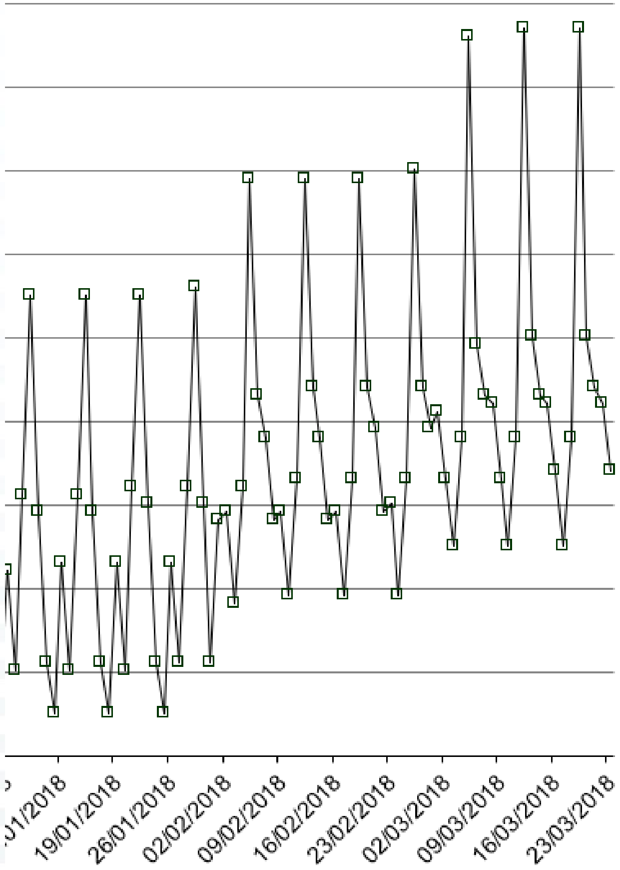
- Averages of equivalent day of previous years captures seasonality better, but struggles with trends and step-changes.

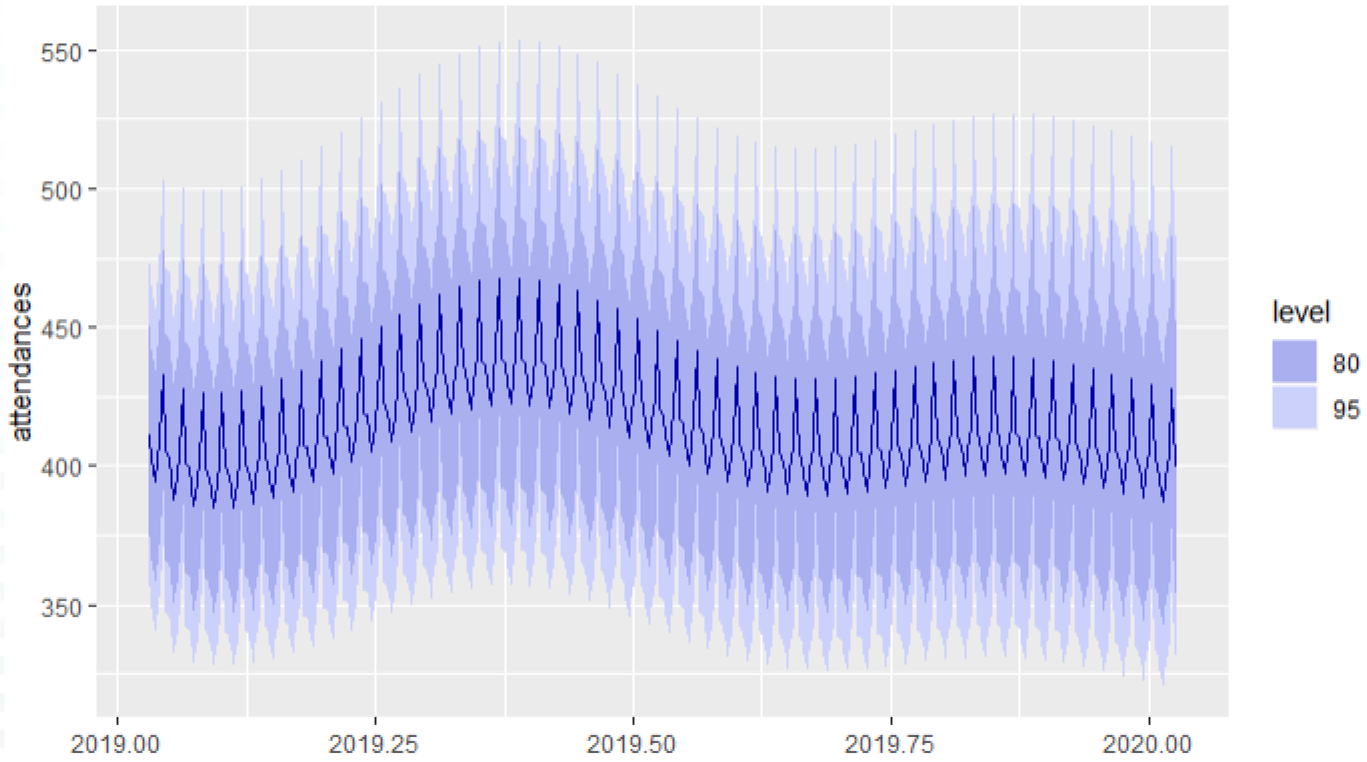
- Considered other 'modern' methods such as Exponential Smoothing (including Holts-Winters), TBATS, etc. These have advantages, including allowing seasonality to vary and giving more weight to recent observations.

- Requirement to include covariates (including dummy variables) meant that a Dynamic Harmonic Regression model with ARIMA errors was obvious choice.

Harmonics - Fourier Terms

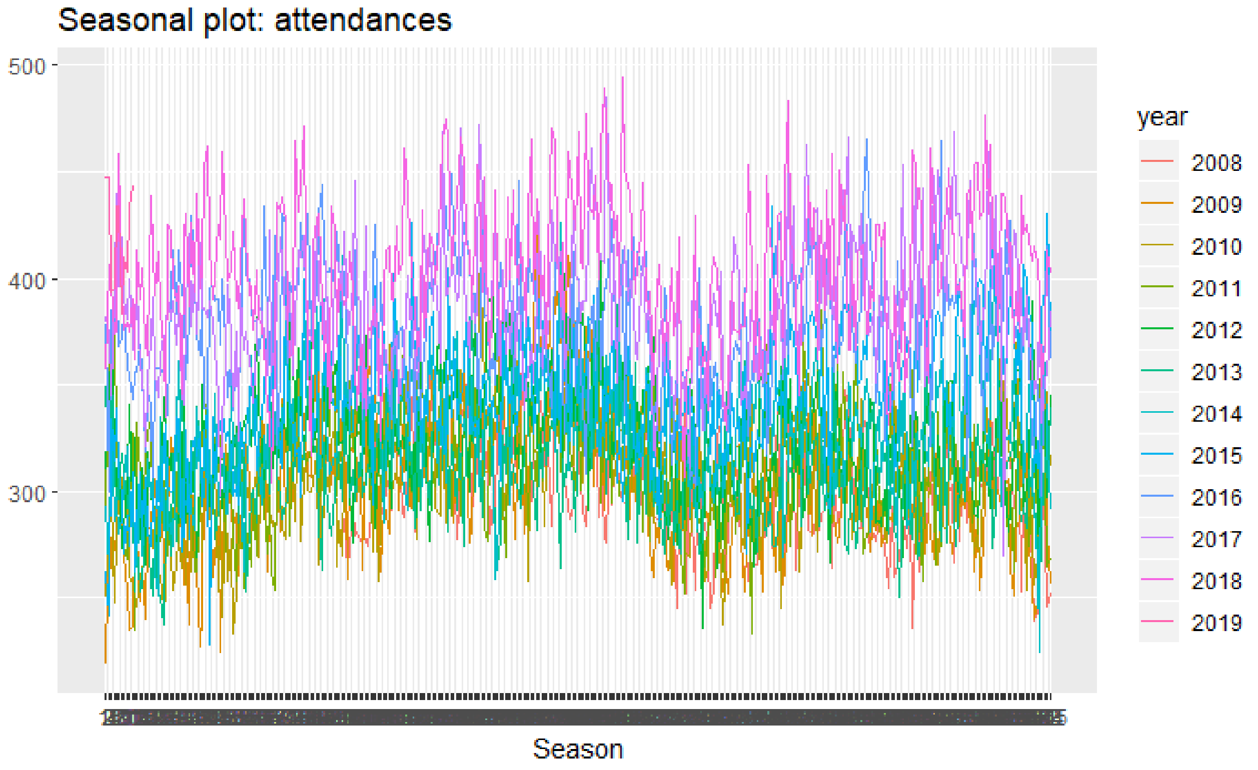
- Models which have dummy variables for each month of the year give jumps from month to month.
- Fourier terms give a smooth seasonal pattern.

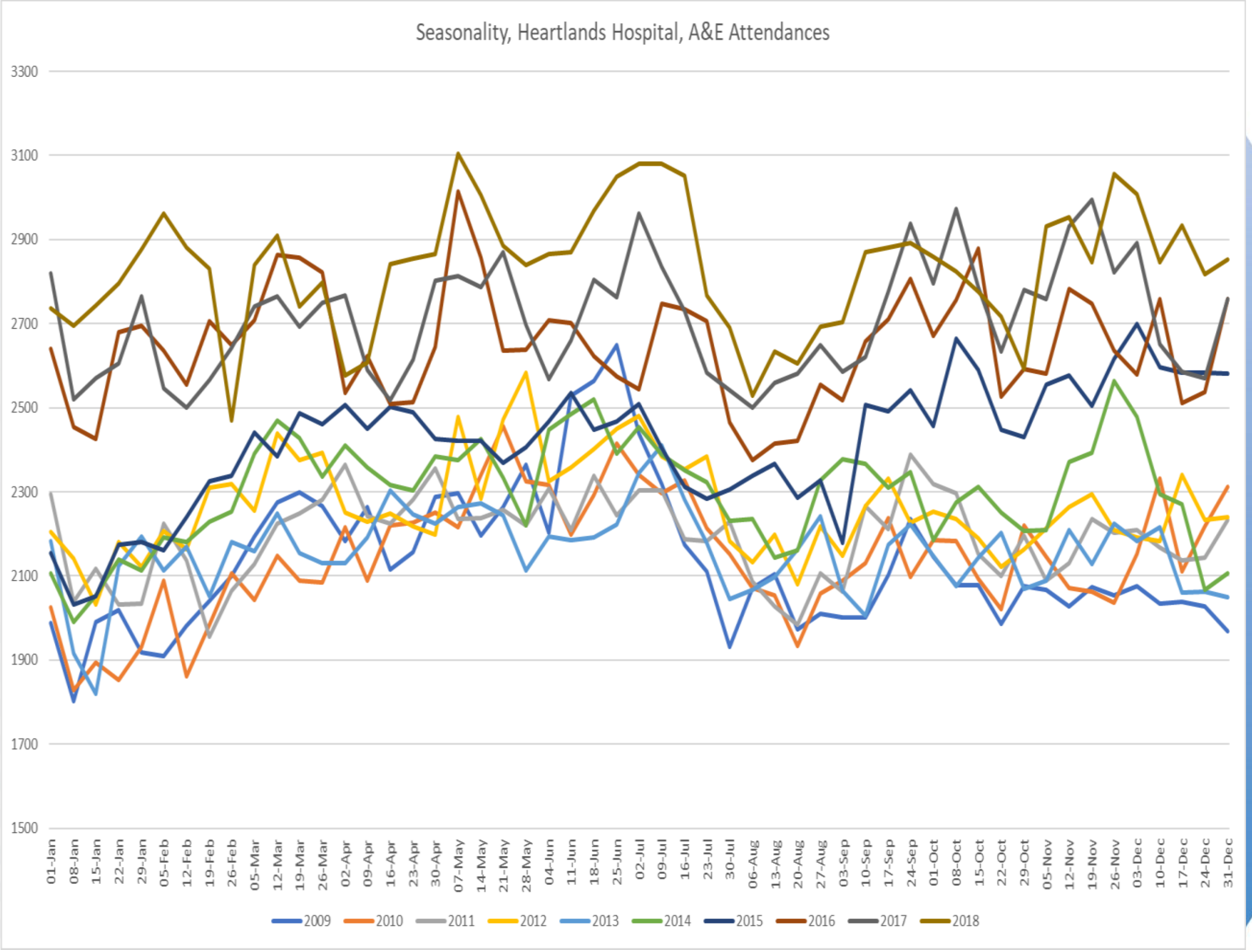




Lots of variability in the data

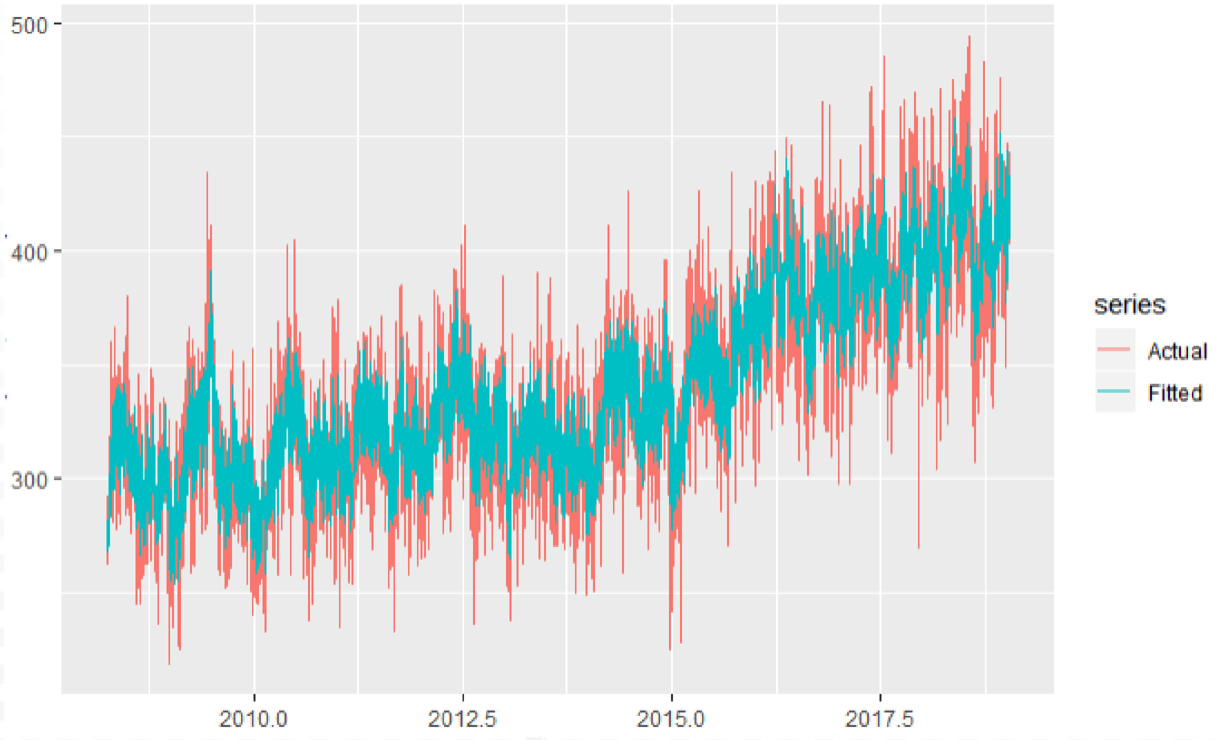
- Mondays not always busiest! Seasonal patterns change, Easter moves, there can be trends and step-changes

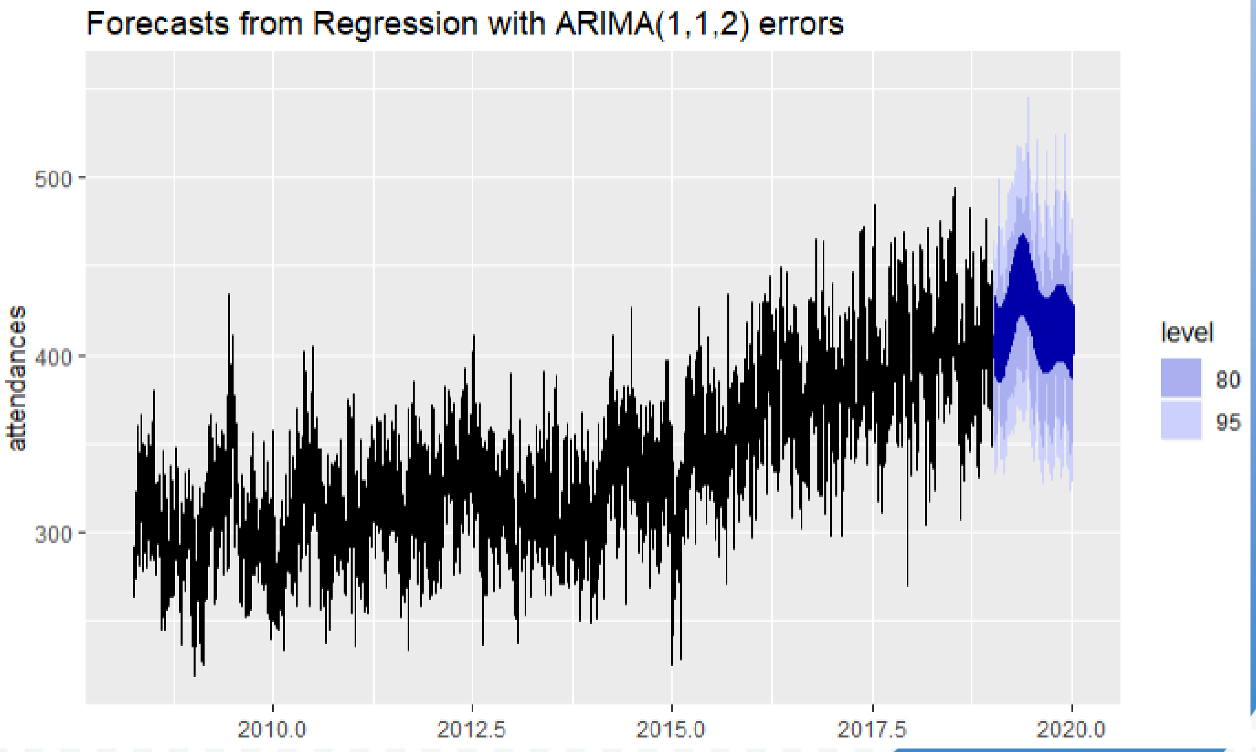




Initial investigation using pure time-series ARIMA models

- Basic auto.arima fits quite well (6.1% MAPE) but does not capture seasonality, giving a flat medium-term forecast.
- Adding fourier terms for day of week and 2 pairs of terms for time of year seasonality gives a better fit (5.6% MAPE) and more useful





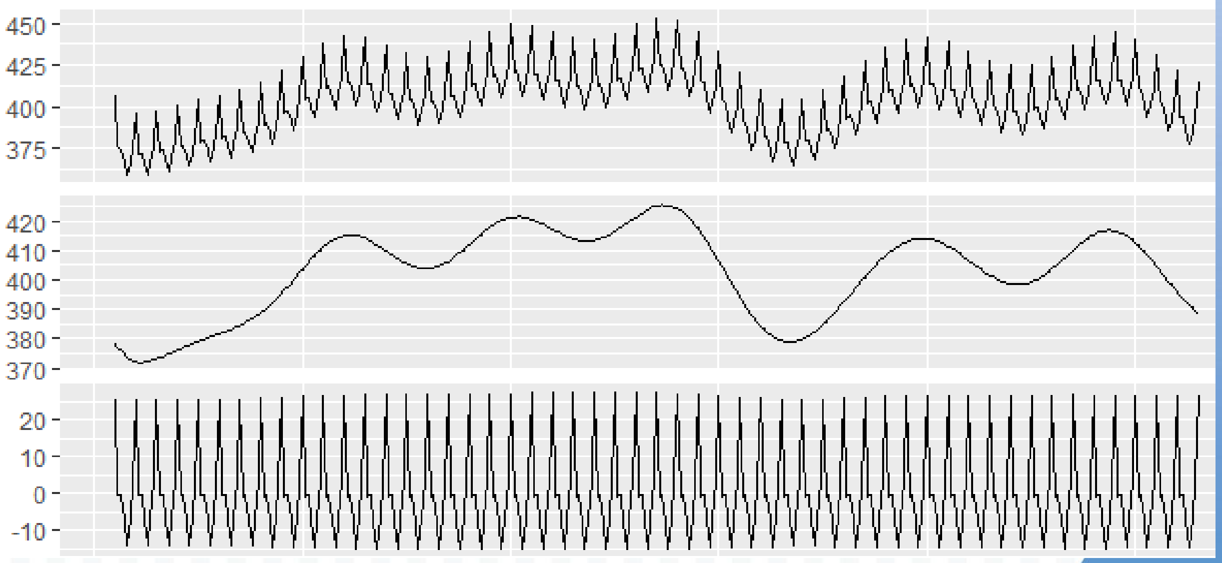
More fourier terms gives a better fit and forecast …

Using a loop in R to select the optimal number of fourier terms leads to a model with 7 pairs of terms, fits better (MAPE 5.5%, AICC -7190.87).

```
for (i in seq(40)) {
  fit <- auto.arima(attendancests, seasonal=FALSE, lambda=0,
         xreg=cbind(DHR03aDummy[1:lastactual,],fourier(attendancests, K=i)))

  print(paste("K=",i,"   AICC=",round(fit[["aicc"]],2)))
}
```

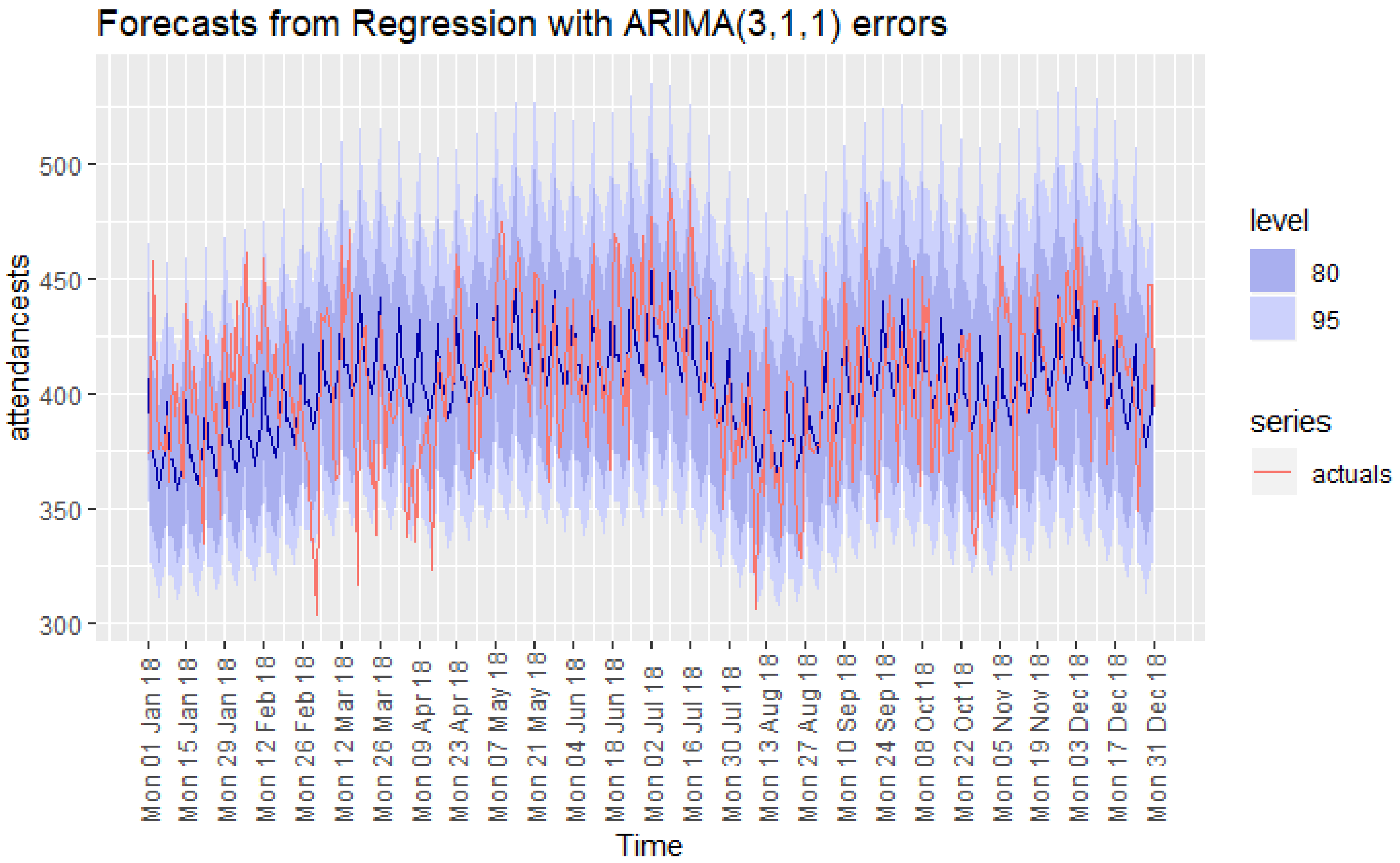
```
[1] "K= 1    AICC= -7095.4"
[1] "K= 2    AICC= -7135.67"
[1] "K= 3    AICC= -7145.78"
[1] "K= 4    AICC= -7160.31"
[1] "K= 5    AICC= -7179.87"
[1] "K= 6    AICC= -7189.95"
[1] "K= 7    AICC= -7190.87"
[1] "K= 8    AICC= -7190.62"
[1] "K= 9    AICC= -7189.69"
[1] "K= 10    AICC= -7188.81"
[1] "K= 11    AICC= -7189.09"
[1] "K= 12    AICC= -7186.27"
[1] "K= 13    AICC= -7185.26"
[1] "K= 14    AICC= -7192.24"
[1] "K= 15    AICC= -7193.54"
```



… but hard to explain effect of e.g. school holidays

This forecasts 17% better than the existing method (forecast MAPE of 6.2% vs 7.5%) and appears to be picking up effects such as school holidays.

However, it is hard to explain and will not deal with moveable holidays like Easter.



Decided better to create dummies for Bank/School holidays

```
DHR03bDummy["Dec_24"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="24/12")
DHR03bDummy["Dec_25"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="25/12")
DHR03bDummy["Dec_26"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="26/12")
DHR03bDummy["Dec_27"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="27/12")
DHR03bDummy["Dec_31"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="31/12")
DHR03bDummy["Jan_01"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="01/01")
DHR03bDummy["Jan_02"]    <-as.numeric(format(SeasDataOneComplete$DateMain,'%d/%m')=="02/01")
```

School holidays started as a list for a region from published websites, but would like to generalise them. E.g. West Midlands October half-term seems to be based on the last Wednesday in October, in other regions it is based on the last Friday in October.

```
LastWedOct <- subset(SeasDataOneComplete$DateMain[1:(lastactual+365)],
                      format(SeasDataOneComplete$DateMain,'%a')=='Wed'
                    & format(SeasDataOneComplete$DateMain,'%d')>='25'
                    & format(SeasDataOneComplete$DateMain,'%b')=='Oct')
OctWe<-LastWedOct-2*24*60*60                                               ## The monday of that  week
DaysMatrix<-outer(SeasDataOneComplete$DateMain[1:(lastactual+365)], OctWe, "-")/(24*60*60)  ## Number of days away from star
t of Oct hol
DaysfromOctWe<-rowSums(DaysMatrix * (col(DaysMatrix) == max.col(-abs(DaysMatrix))))    ## Number of days away from closest f
irst Mon in Sep
AllDummy[,"OctWeWE_0"]<-rowSums(outer(DaysfromOctWe,seq(-2,-1,1),"==")*1)
AllDummy[,"OctWeWkp0"]<-rowSums(outer(DaysfromOctWe,seq( 0, 4,1),"==")*1)
AllDummy[,"OctWeWEp0"]<-rowSums(outer(DaysfromOctWe,seq( 5, 6,1),"==")*1)
```

Log-linear relationship means effects are multiplied together

```
```{r}
KTerms<-1
DHRTest <- auto.arima(attendancests, seasonal=FALSE, lambda=0,
          xreg=as.matrix(cbind(UseDummy[1:lastactual,],fourier(attendancests, K=KTerms))))
summary(DHRTest)
```
```

```
Series: attendancests
Regression with ARIMA(1,1,3) errors
Box Cox transformation: lambda= 0

Coefficients:
         ar1      ma1     ma2     ma3  drift  Monday  Tuesday  Wednesday  Thursday  Friday  Saturday   Dec_24   Dec_25  Dec_26  Dec_27   Dec_31  Jan_01  Jan_02  Sch1_Feb
      0.9228  -1.6389  0.5412  0.1026  1e-04  0.2196   0.1042     0.0915    0.0868  0.0843   -0.0015  -0.0449  -0.3005  0.0116  0.1463  -0.0304  0.0459  0.1302   -0.0354
s.e.  0.0224   0.0293  0.0356  0.0203  1e-04  0.0040   0.0043     0.0043    0.0043  0.0043    0.0038   0.0215   0.0221  0.0221  0.0215   0.0215  0.0222  0.0216    0.0106
      Sch1_Eas  Sch1_Xms  BHSprMo  BHSprSa  BHSprSu  BHSprTu  BHMayMo  BHMaySa  BHMaySu  BHMayTu  BHEasMo  BHEasSa  BHEasSu  BHEasTu  BHEasFr  MayWkp0  MayWEp1  SepWE_6  SepWk_6
       -0.0012   -0.0089  -0.1682  -0.0209  -0.0098   0.0658  -0.1490   0.0269   0.0069   0.0901  -0.1924   0.0550  -0.0435   0.0692  -0.0881  -0.0485   0.0030  -0.0333  -0.0503
s.e.    0.0100    0.0104   0.0245   0.0213   0.0220   0.0232   0.0216   0.0212   0.0217   0.0211   0.0221   0.0223   0.0223   0.0213   0.0214   0.0153   0.0171   0.0174   0.0144
      SepWE_5  SepWk_5  SepWE_4  SepWk_4  SepWE_3  SepWk_3  SepWE_2  SepWk_2  SepWE_1  SepWk_1  SepWE_0  SepWkp0  OctWeWE_0  OctWeWkp0  OctWeWEp0    S1-365   C1-365
      -0.0111  -0.0653  -0.0847  -0.1016  -0.0527  -0.0764  -0.0545  -0.0675  -0.0015  -0.0727  -0.0401  -0.0373     -0.023    -0.0586     0.0341   -0.0021  -0.0175
s.e.   0.0192   0.0158   0.0199   0.0164   0.0202   0.0166   0.0201   0.0163   0.0197   0.0156   0.0188   0.0137      0.017     0.0130     0.0170    0.0078   0.0077

sigma^2 estimated as 0.003726:  log likelihood=4050.08
AIC=-7988.17   AICc=-7985.94   BIC=-7653.31

Training set error measures:
                    ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
Training set 0.6891119 27.84409 22.00764 -0.1766208 4.750368 0.3954737 0.00157109
```

lambda=0 means log-linear relationship

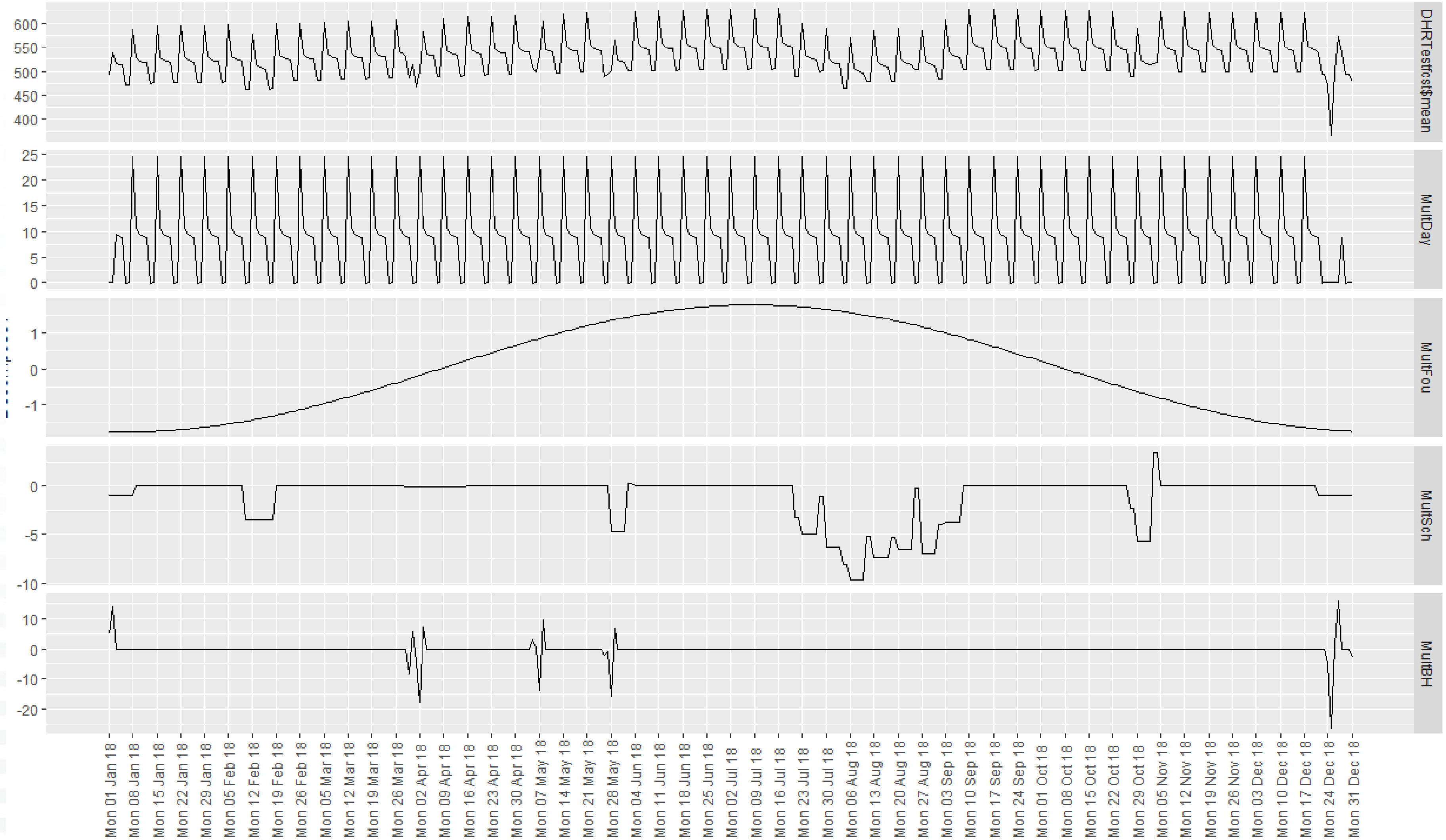
$$\log(y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ \ldots$$

$$y = e^{(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ \ldots)}$$

$$y = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2} \ \ldots$$

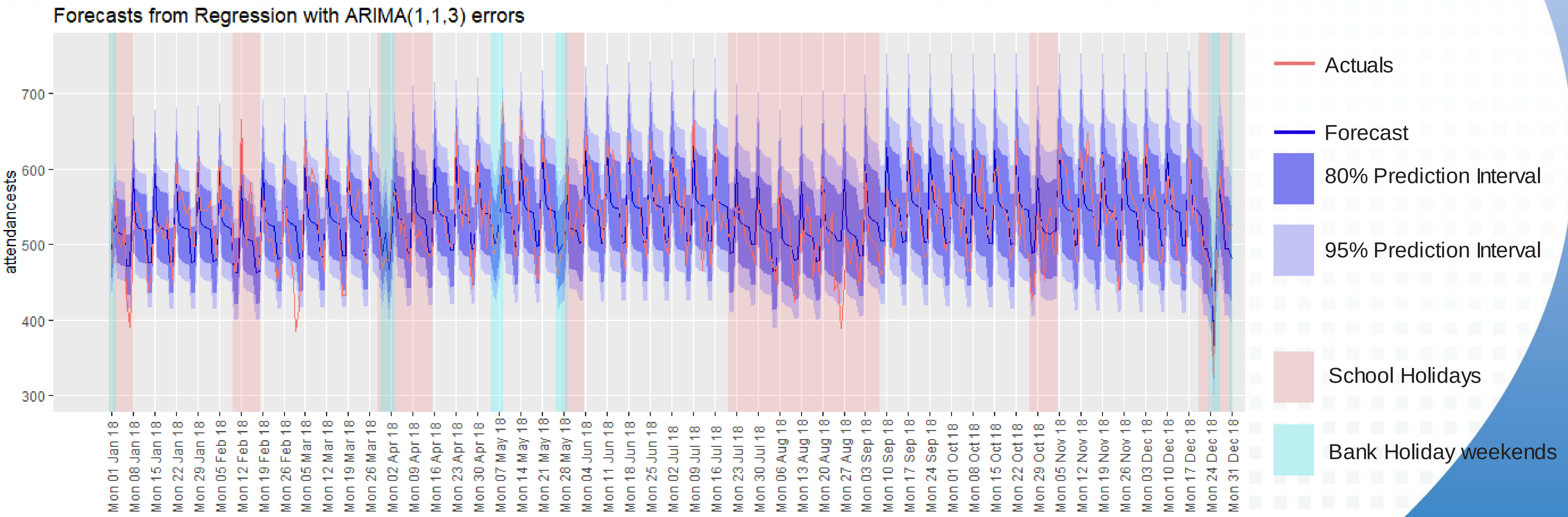
So, for example, Monday co-efficient of 0.2196 means that attendances on an average Monday are $e^{0.2196} = 1.2456$ times attendances on a Sunday, i.e. 24.56% higher.

Separate effects of dummy variables and single fourier pair for time of year



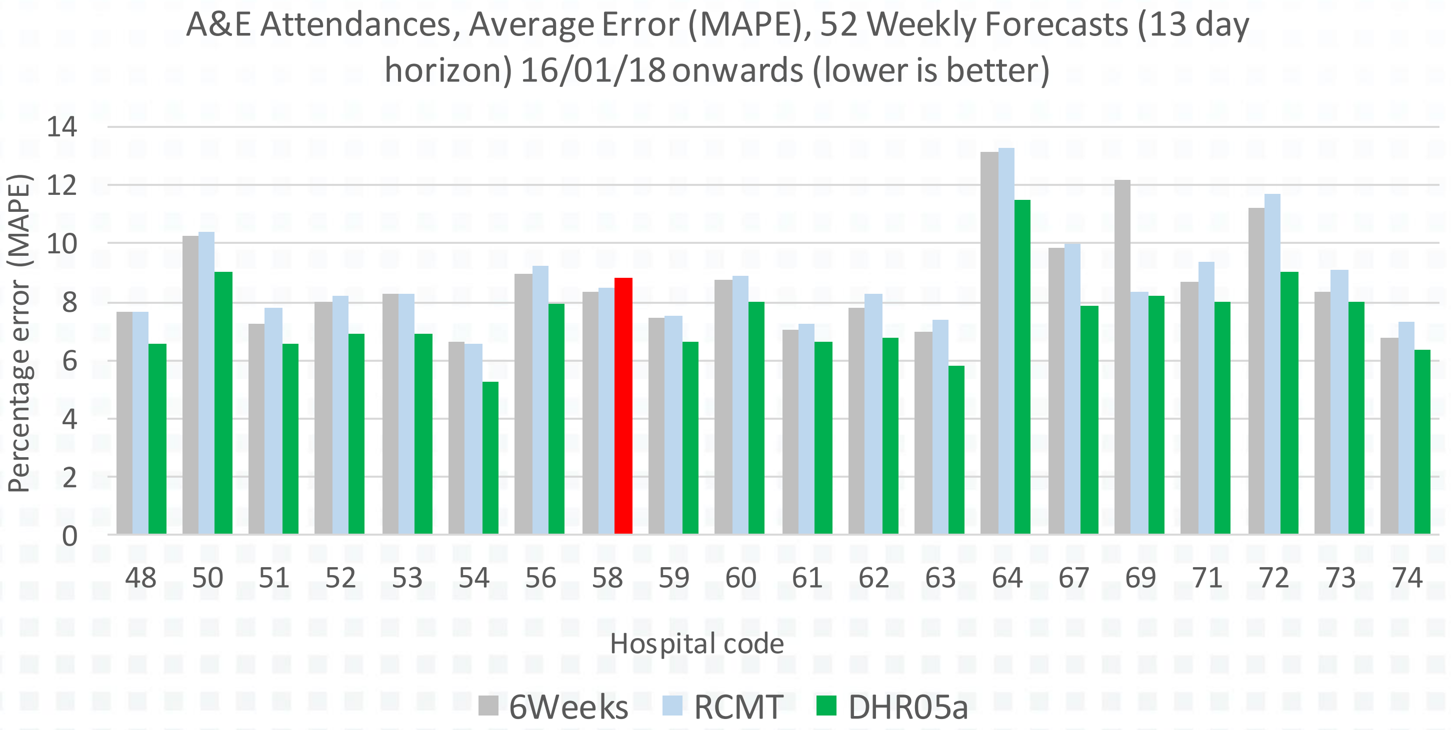
Can use annotate in R to overlay the dummy variables on a chart

```
autoplot(DHRTestfcst, include=0) +
 autolayer(actuals) +
  scale_x_continuous(breaks=Abreaks,labels=Adates, minor_breaks = NULL) +
  annotate(geom="rect",xmin=DateDumSch-0.5/Freq, xmax=DateDumSch+0.5/Freq, ymin=-Inf, ymax=Inf, alpha=0.1, fill="red") +
  annotate(geom="rect",xmin=DateDumBH -0.5/Freq, xmax=DateDumBH +0.5/Freq, ymin=-Inf, ymax=Inf, alpha=0.2, fill="cyan") +
  theme(axis.text.x=element_text(angle=90, hjust=0, vjust=0.5)) +
  theme(panel.grid.minor = element_line(color = "grey")) +
  scale_y_continuous(minor_breaks=NULL)
```



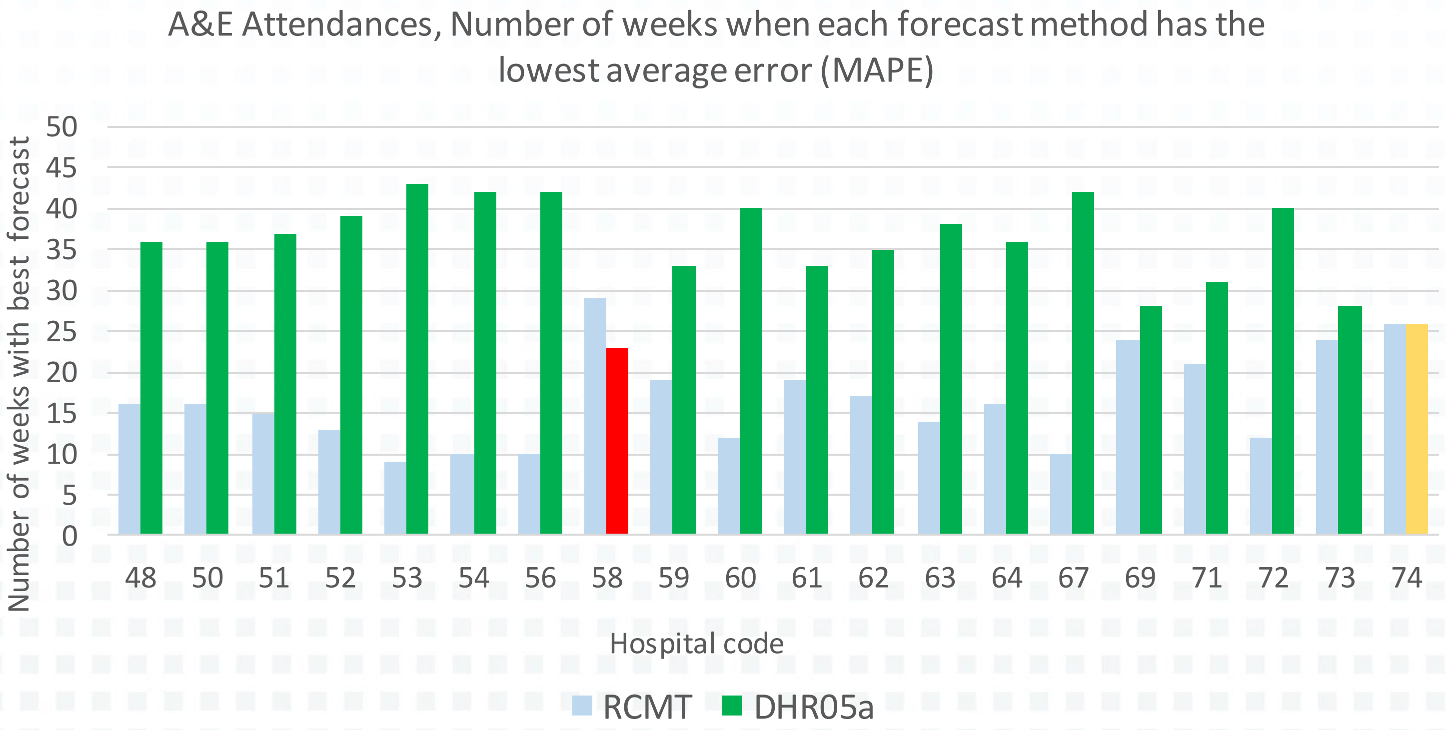
Separate R script to run on server and evaluate many forecasts

Calculating 52 weekly forecasts from early 2018 onwards, using the new method outperforms (by a relatively small margin) the existing weekly method for all hospitals apart from 58, where we know that the inclusion of a Walk in Centre affects the data.



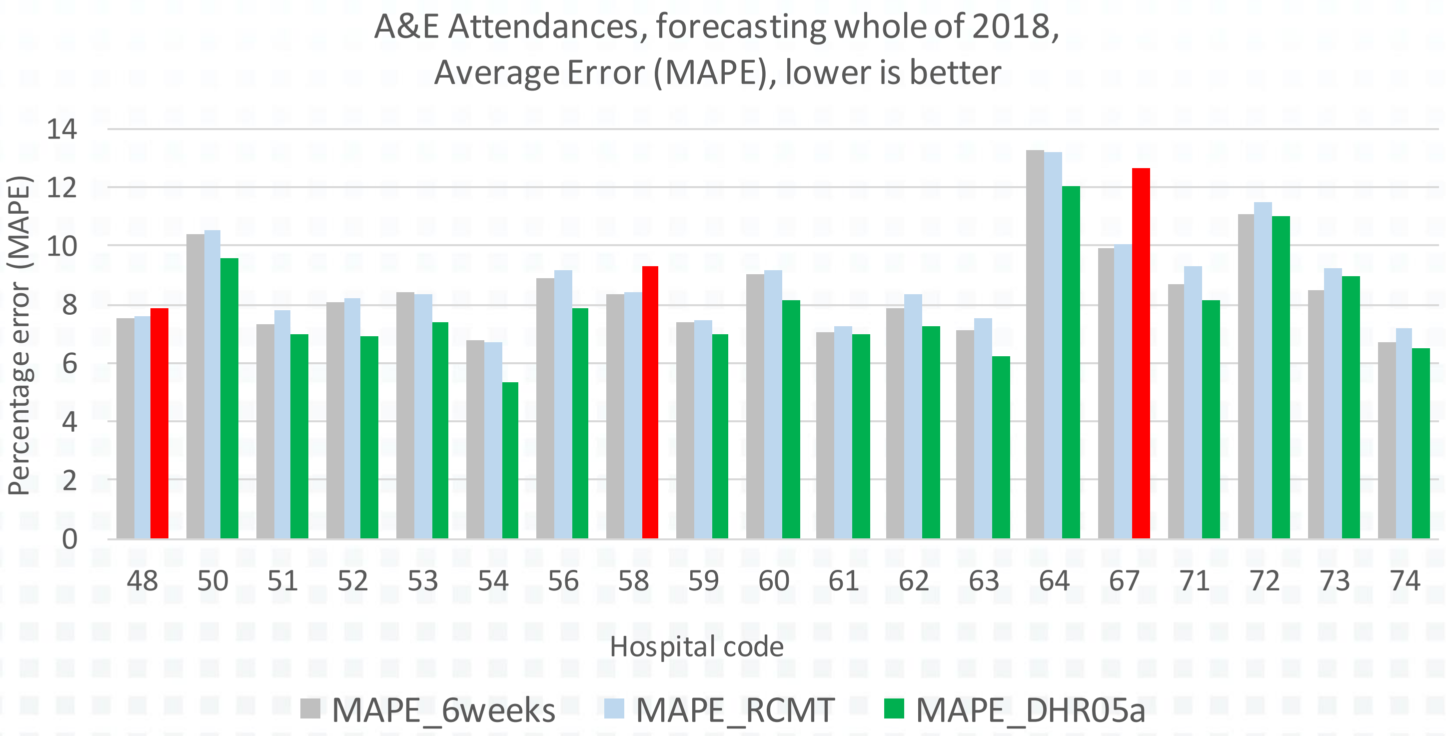
Weekly Performance-attendances

There are still some weeks when the existing forecast was closer to the actuals than the new method. E.g. for hospital 74, the existing and the new method each won 50%.



Annual Performance-attendances

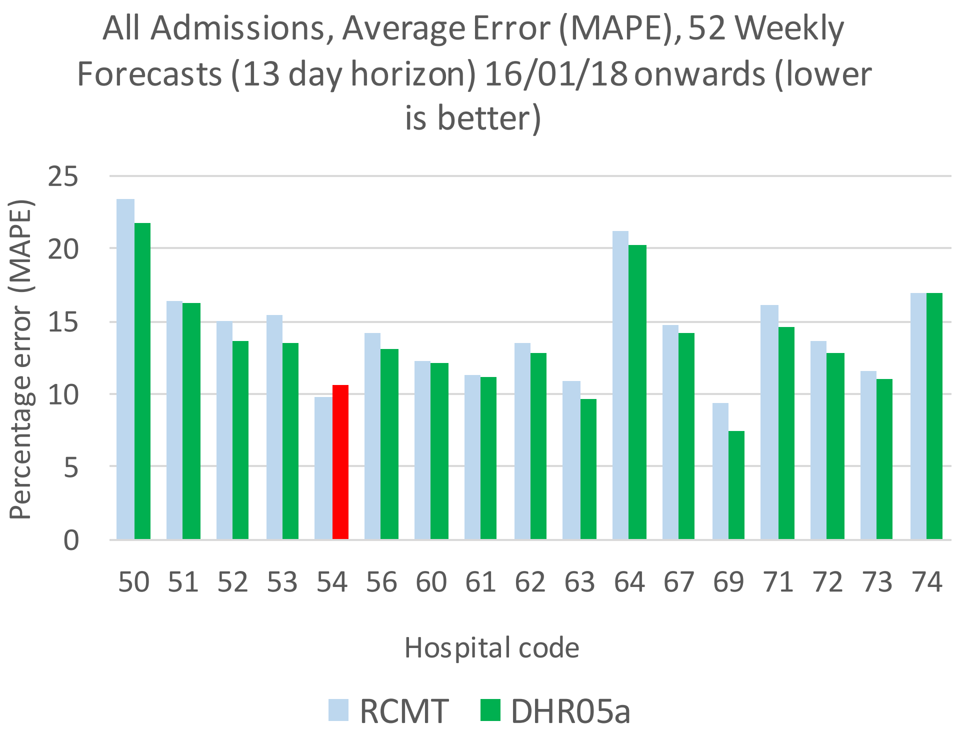
The new method allows for longer forecast horizons. Even forecasting the whole of 2018 using only data to end-2017, the new method outperforms the RCMT method for 80% of hospitals, even though the RCMT uses data up to the end of each week.

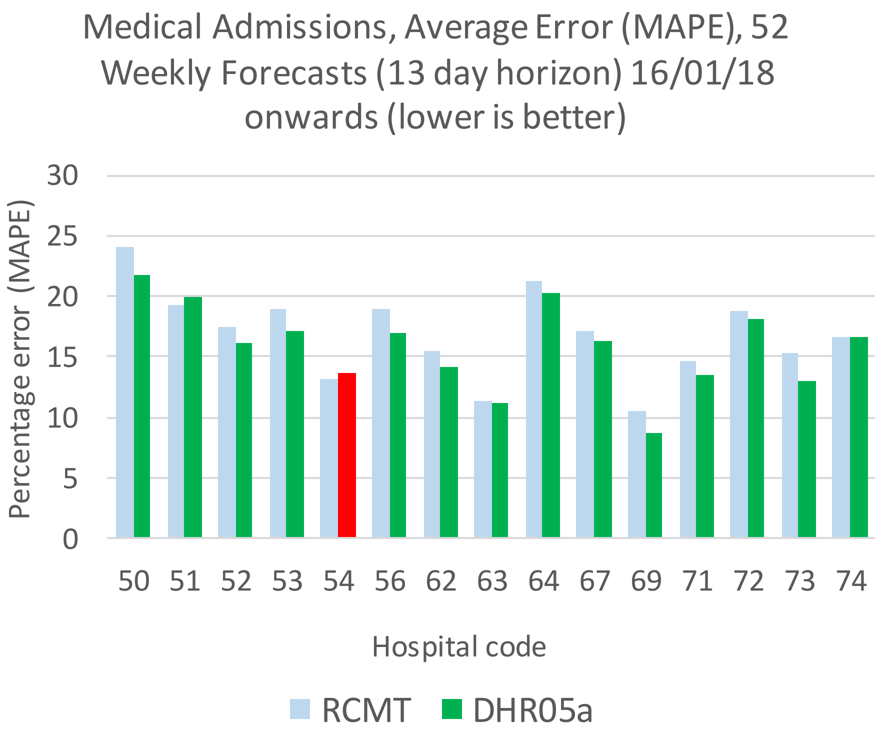


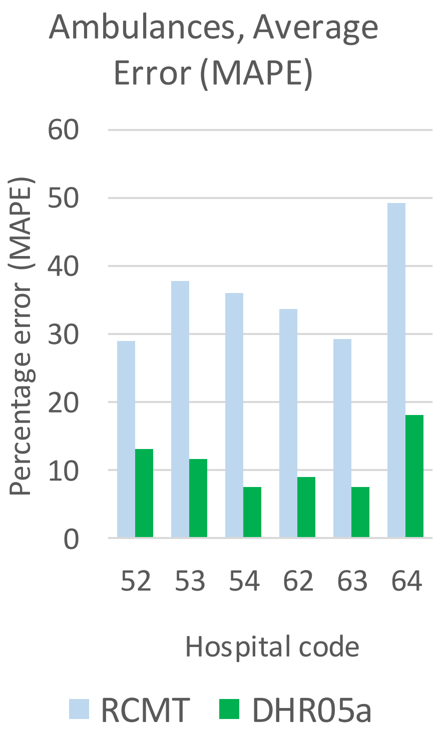
Note: (Hospital 69 not shown on the chart as the MAPE was very high)

Weekly Performance-admissions/ambulances

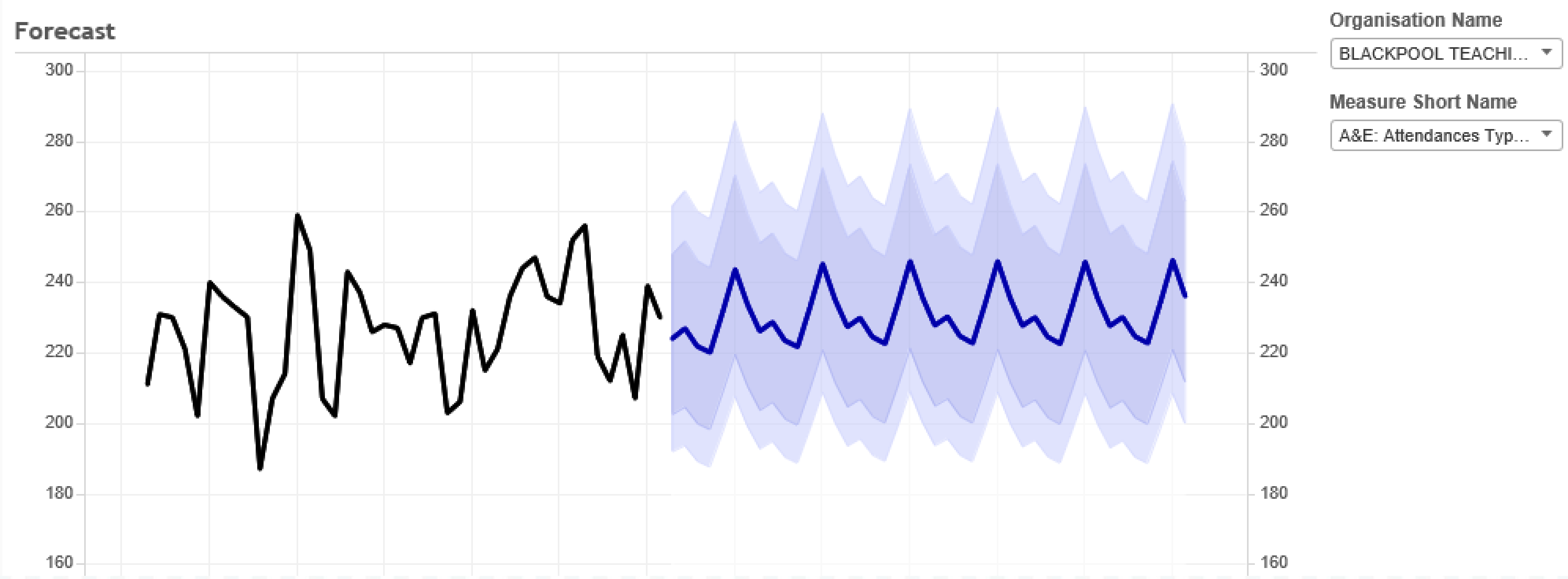
Forecast performance for 2018 data is also better for admissions and ambulances for all except Hospital 54





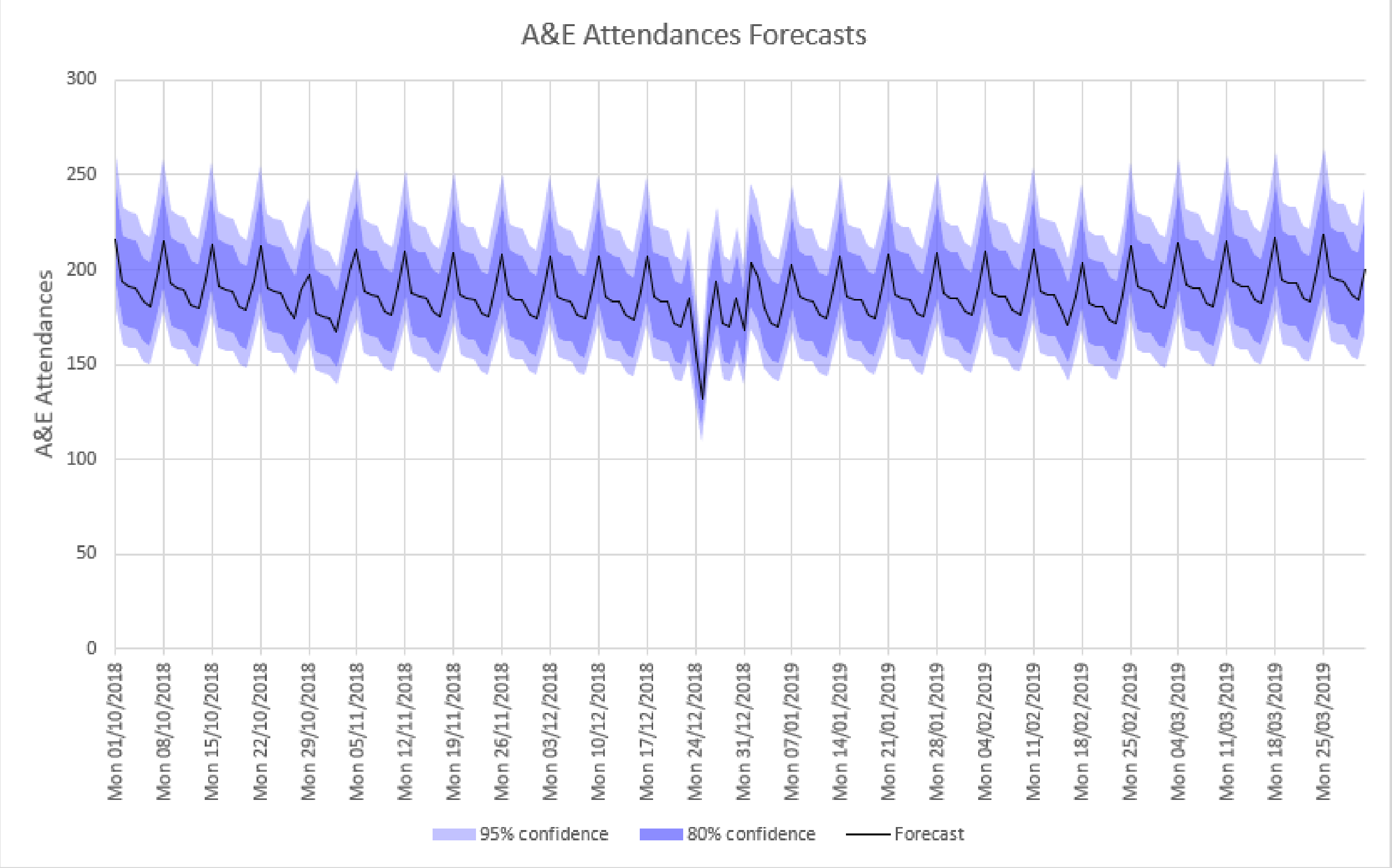


Scheduled R script outputs to DW, can present in e.g. Tableau



Or in Excel

PRH Winter Attendances Projections						
Day	Date	95% lower limit	80% lower limit	Forecast	80% upper limit	95% upper limit
Monday	01/10/2018	180	191	216	244	260
Tuesday	02/10/2018	161	172	194	219	233
Wednesday	03/10/2018	159	170	191	216	230
Thursday	04/10/2018	159	169	191	215	230
Friday	05/10/2018	152	162	183	206	220
Saturday	06/10/2018	150	160	181	204	218
Sunday	07/10/2018	163	174	196	221	236
Monday	08/10/2018	178	190	215	242	258
Tuesday	09/10/2018	160	171	193	217	232
Wednesday	10/10/2018	158	169	190	215	229
Thursday	11/10/2018	158	168	190	214	228
Friday	12/10/2018	151	161	182	205	219
Saturday	13/10/2018	149	159	180	203	216
Sunday	14/10/2018	162	173	195	220	235
Monday	15/10/2018	177	189	214	241	257
Tuesday	16/10/2018	159	170	191	216	230
Wednesday	17/10/2018	157	168	189	214	228
Thursday	18/10/2018	157	167	189	213	227
Friday	19/10/2018	150	160	181	204	218
Saturday	20/10/2018	149	158	179	202	215
Sunday	21/10/2018	161	172	194	219	234
Monday	22/10/2018	176	188	212	240	256
Tuesday	23/10/2018	158	169	190	215	229
Wednesday	24/10/2018	156	167	188	213	227
Thursday	25/10/2018	156	166	188	212	226

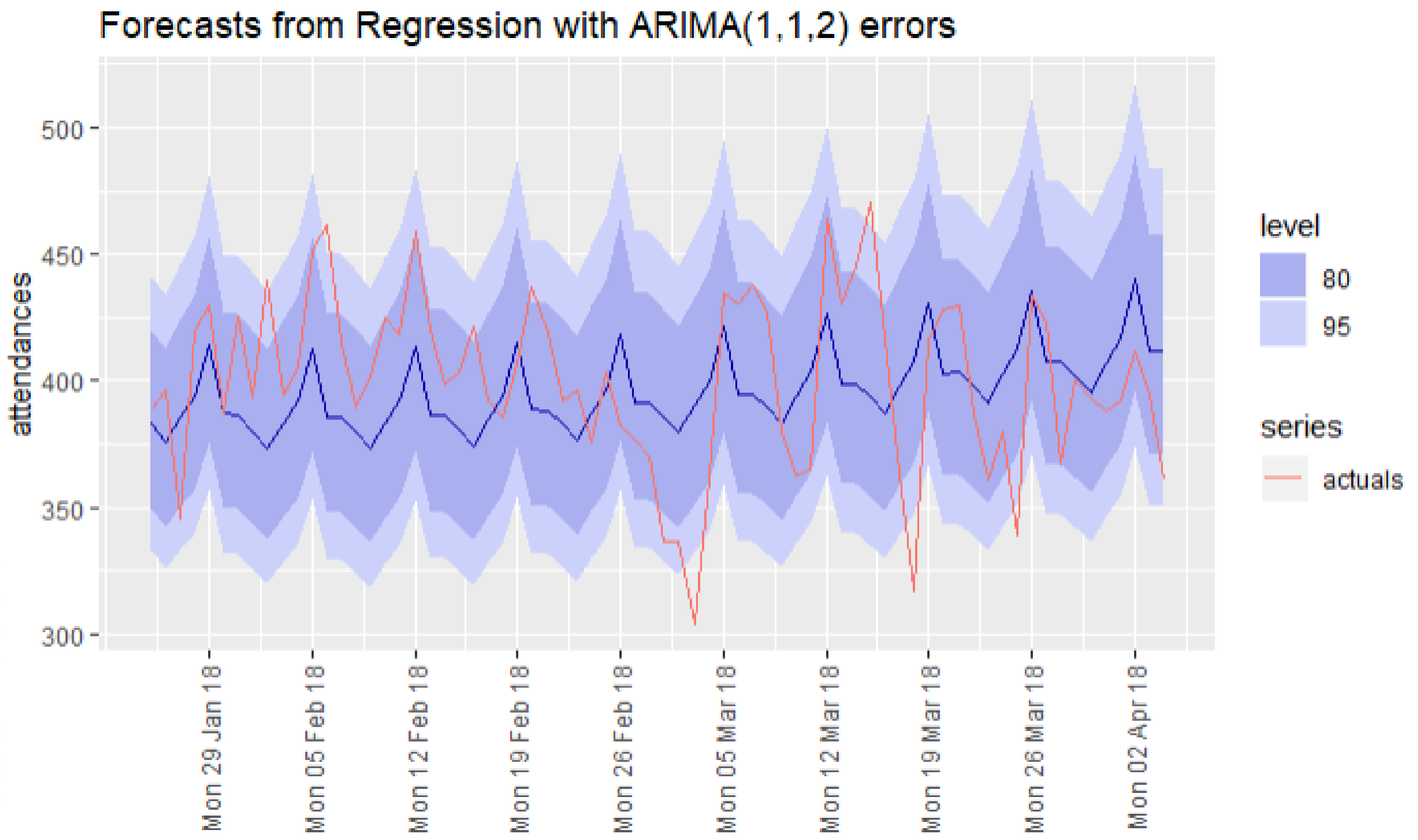


Example 10 week forecast with confidence limits

- Example forecast vs actuals

- Clear Monday peaks, as expected. For this hospital, Fridays have the lowest attendances (elsewhere, have seen Saturday as lowest)

- 79% of actuals are within 80% limits and 91% are within 95% limits



Intended next steps

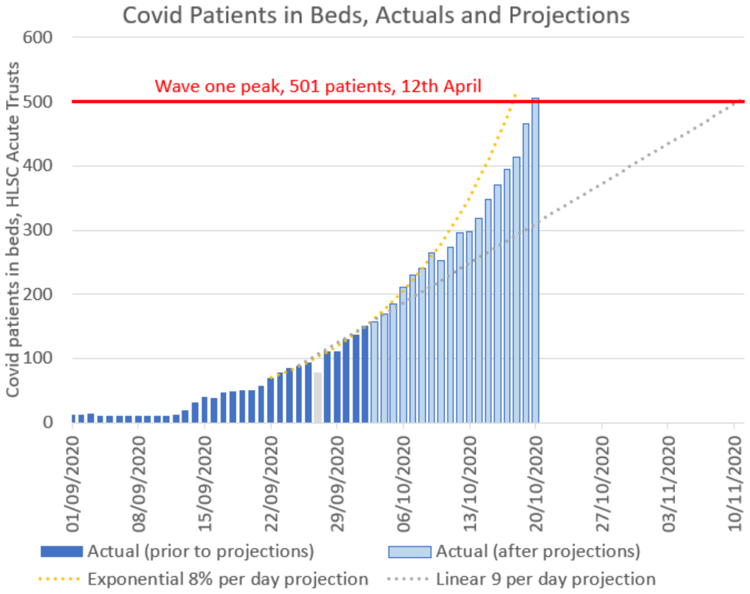
- Generalise school holiday dummies ideally to not need local lists.
- Incorporate effects of weather rather than fourier terms for time of year.

Covid

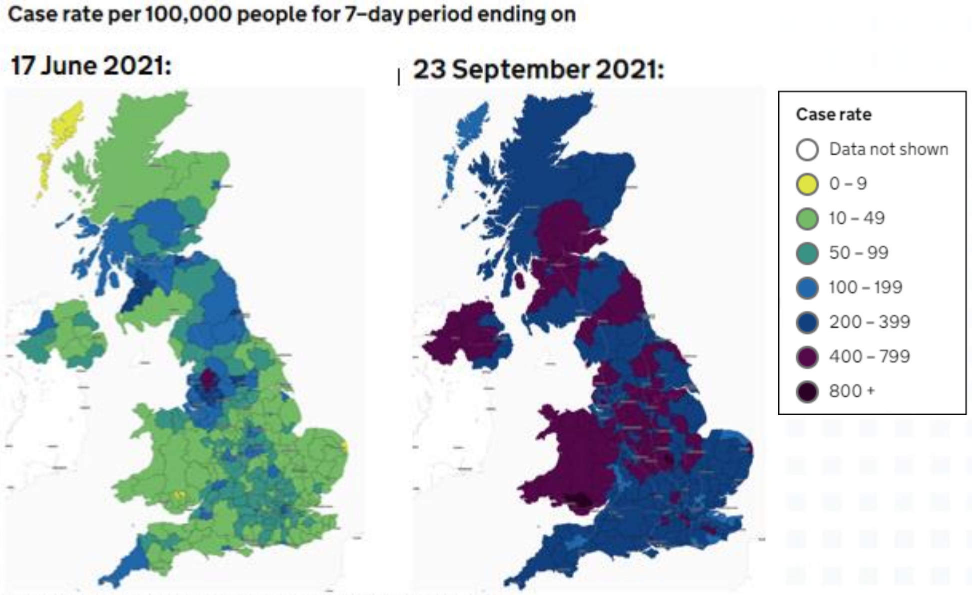
-projections not forecasts

The lessons of the first Covid wave were to understand the power of exponential growth and to look at what is actually happening day to day, as well as modelling.

The lesson of the second and third Covid waves was to look where it is happening first, not at the national figures, because it will soon be happening everywhere.



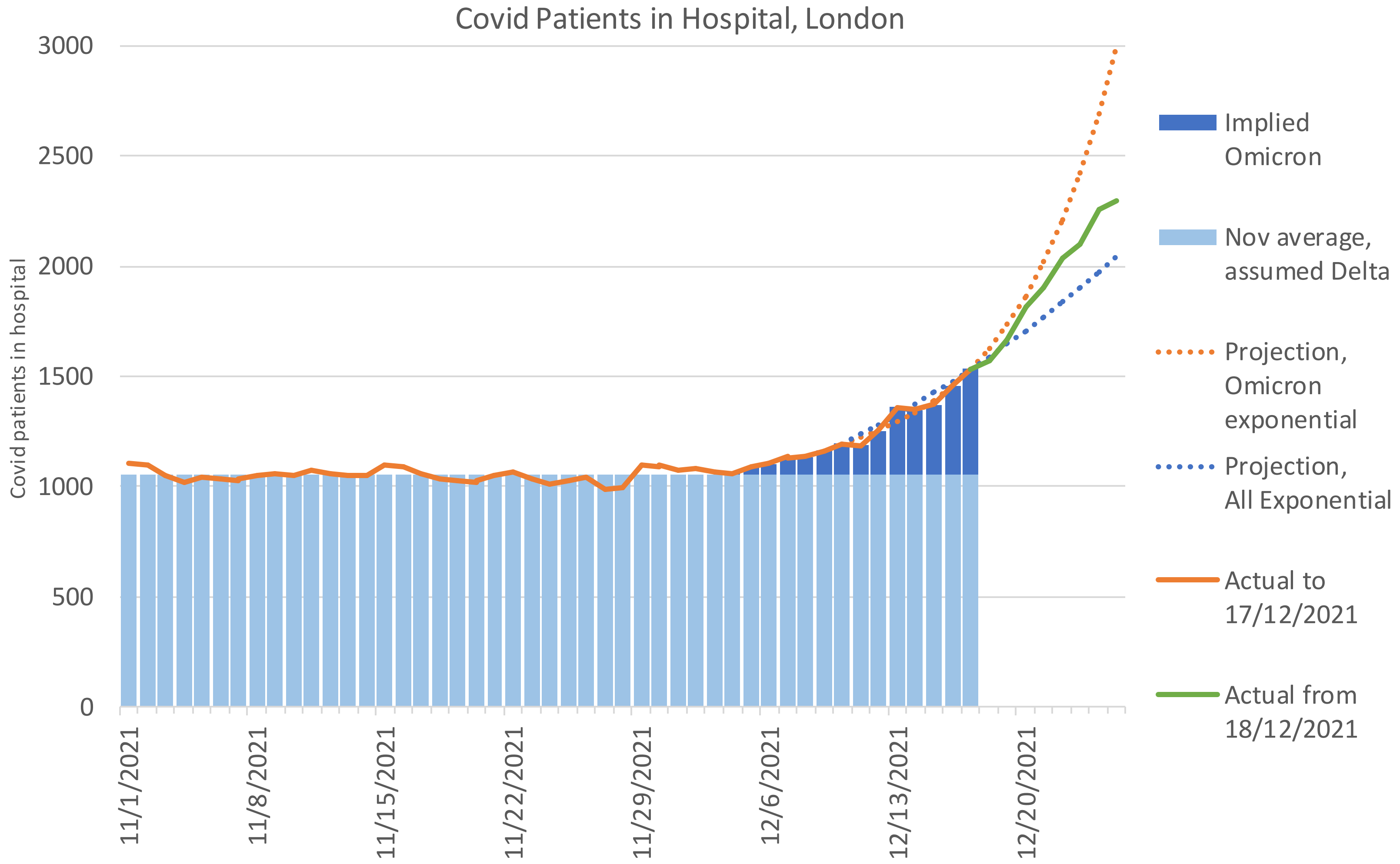
Source: MLCSU from Covid Daily SitRep



Source: https://coronavirus.data.gov.uk/details/interactive-map

Previous Covid waves started from almost zero cases in hospital. The lesson of this wave of Omicron on top of the current Delta is that looking just at the overall growth rate is misleading.

Previous Covid waves started from almost zero cases in hospital. This one is not and so looking just at the overall growth rate is dangerously misleading.



If you saw hospital Omicron cases rising from zero at an average +19% per day, doubling in 4 days, you might project forward like the dotted orange line.

But if you already had 1,000 Delta Covid patients in hospital, it would look very different.

Total Covid patients have an average increase of only +4% per day, doubling in 19 days, so you might project forward like the dotted blue line.

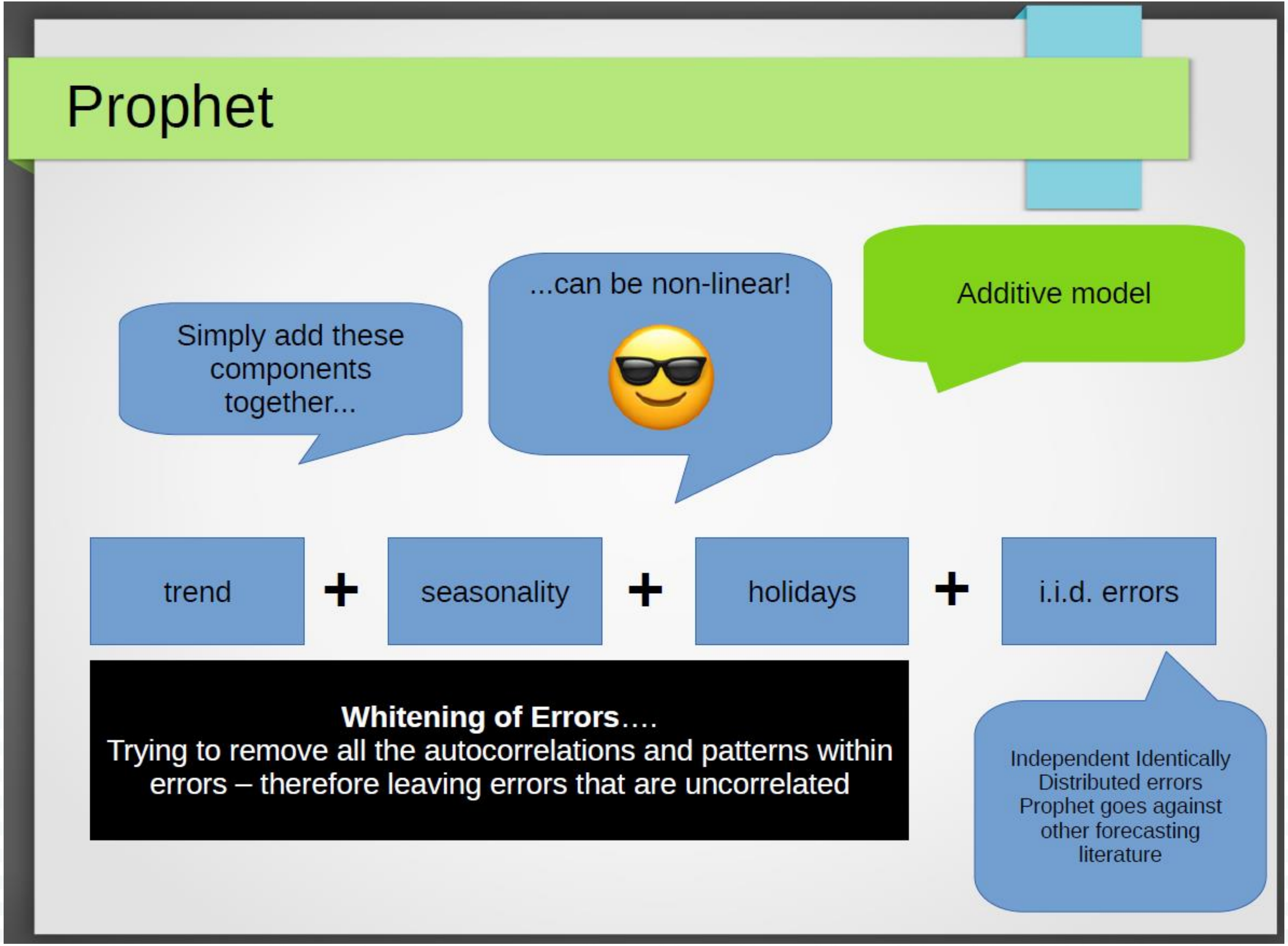
Both look plausible, but they lead to very different outcomes. We believe the dotted orange line is most realistic, rapid Omicron growth on top of stable or falling Delta.

The solid orange line is actual Covid patients in London hospitals to Friday 17/12/2021, which have been increasing for 16 days. The projections were made when this was the latest data available.

The solid green line is actual Covid patients in London hospitals to 25/12/2021. Although the growth was a little less than the orange dotted line, it was markedly higher than the blue dotted line.

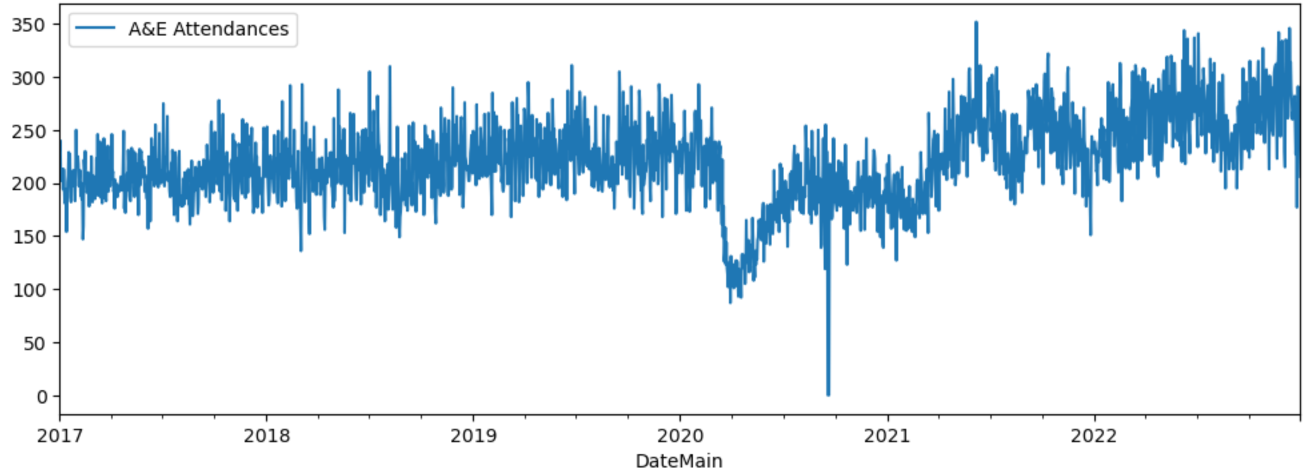
Source: MLCSU from metric hospitalCases from https://coronavirus.data.gov.uk/details/download, downloaded Monday 20/12/2021 pm.
Projections based on growth 10/12/2021-17/12/2021

Prophet
-revisiting daily attendance forecasts



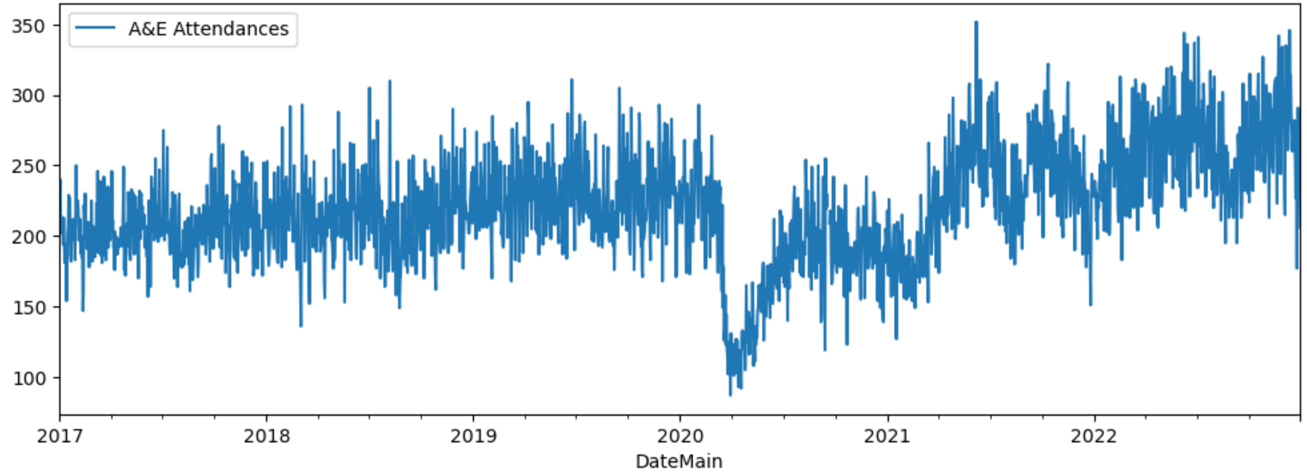
Slide from HSMA5 (Health Service Modelling Associates) training materials (https://github.com/hsma5/9b_forecasting_with_prophet)

Prophet comfortable with missing data




```
[13]: # Prophet handles missing data fine.
      # Should never be genuine zeros in this dataset, so mark the zeroes as NaN
      y_train.replace(0, np.nan, inplace=True)
      y_train.plot(figsize=(12,4))
```

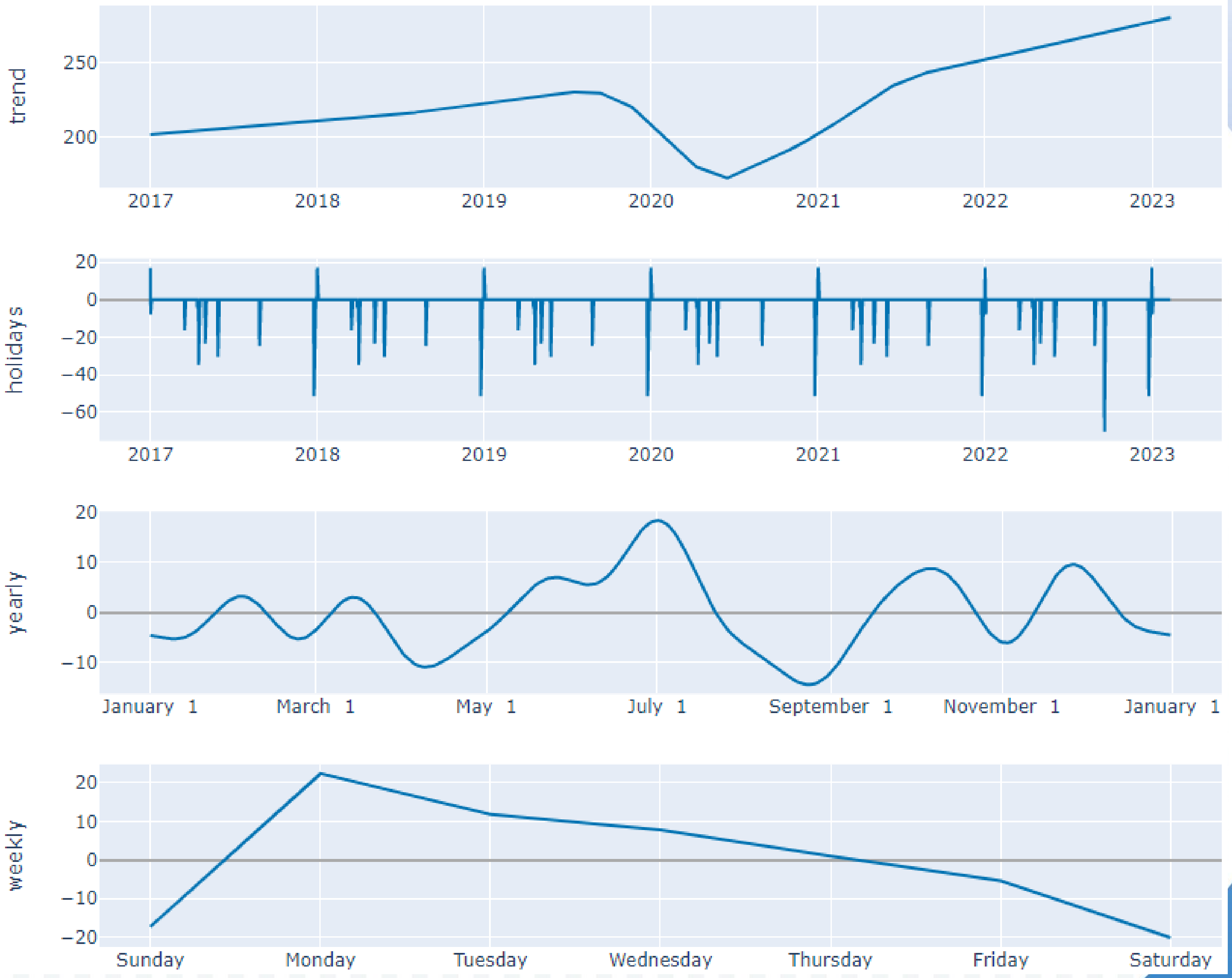
```
[13]: <AxesSubplot:xlabel='DateMain'>
```



Flexible components

- Linear piece-wise Trend
- (Relatively!) easy to add Bank holiday dummies (including moveable ones like Easter)
- Yearly seasonality currently capturing climate and school holiday effects
- Day of week seasonality as expected
- Can add additional regressors

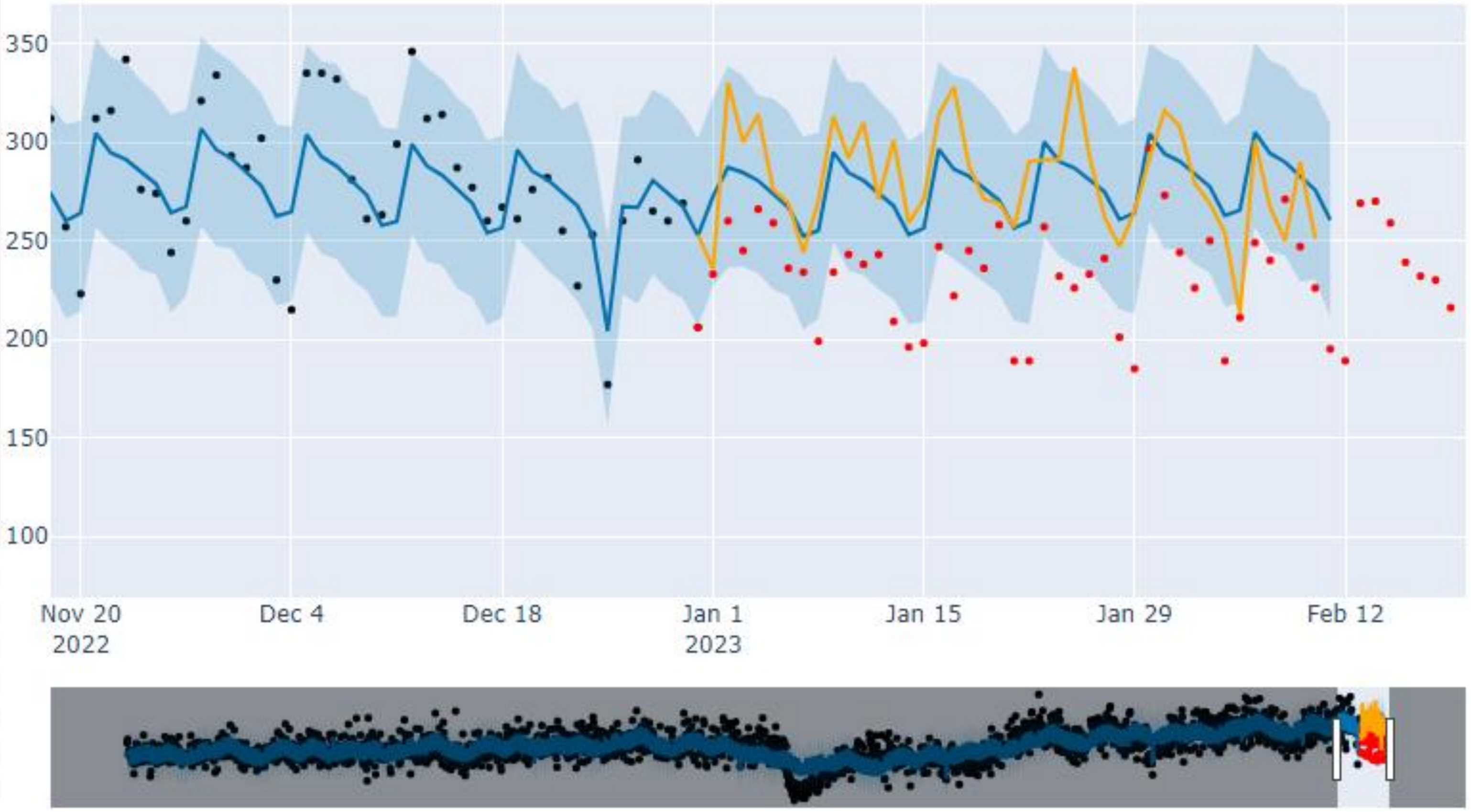
https://nbviewer.jupyter.org/github/nicolasfauchereau/Auckland_Cycling/blob/master/notebooks/Auckland_cycling_and_weather.ipynb



Nice plots, but initially poor performance!

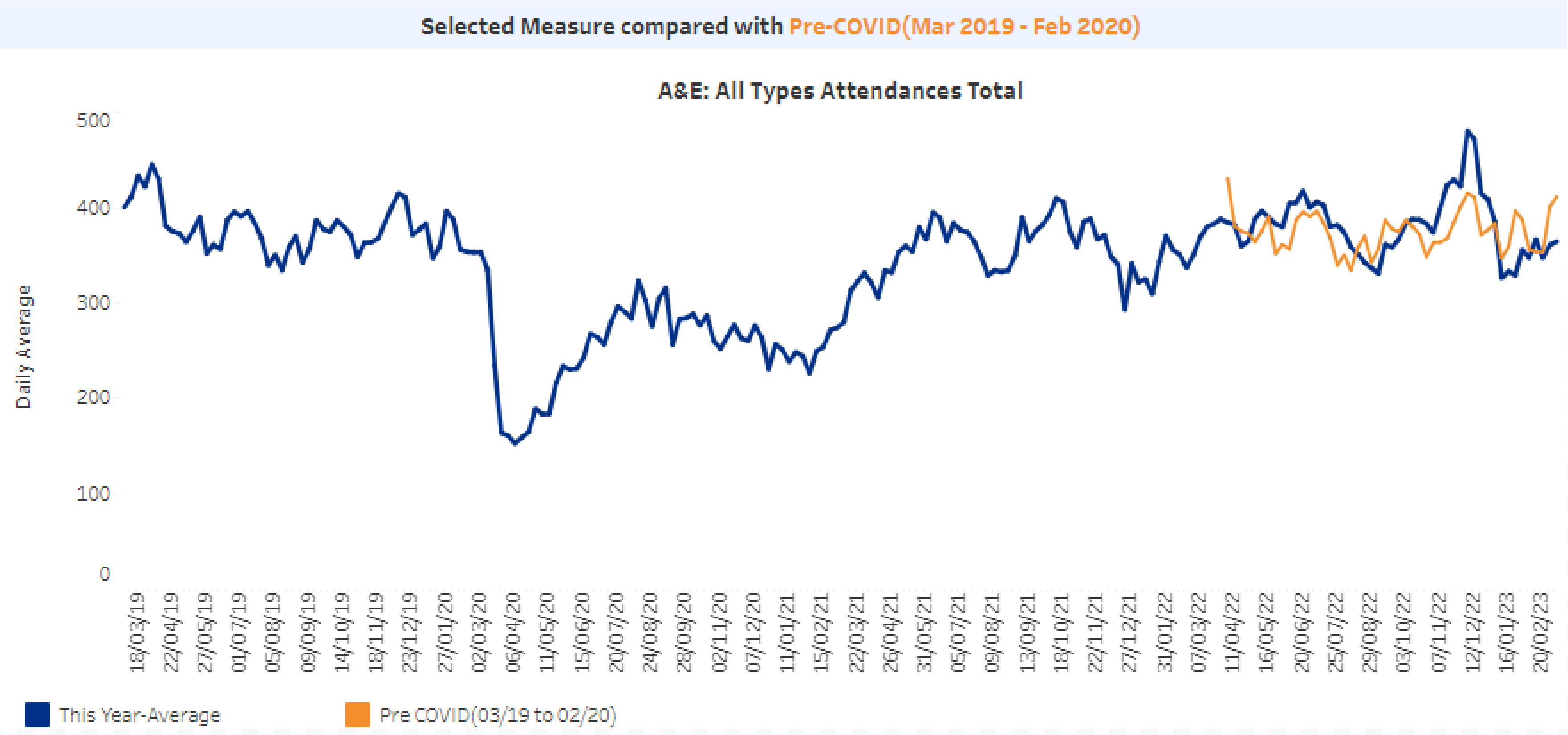
```
# Create the default prophet plot
plotb = plot_plotly(model, prophet_forecast)
# add the recent actuals
plotb.add_trace(go.Scatter(
        name='Recent',
        x=yp_test['ds'],
        y=yp_test['y'],
        marker=dict(color='red', size=4),
        mode='markers'
    ))
# and also the MLCSU original forecast
plotb.add_trace(go.Scatter(
        name='Original Projections',
        x=yp_orig['ds'],
        y=yp_orig['y'],
        mode='lines',
        line=dict(color='orange', width=2)
    ))
# show the main chart as six weeks of training and six weeks of forecast
# (slider underneath shows whole period)
showfrom = datetime.datetime.strptime('18/11/2022', "%d/%m/%Y")
showto   = datetime.datetime.strptime('20/02/2023', "%d/%m/%Y")
plotb.update_xaxes(range=[showfrom,showto])

plotb.show()
```




```
MAPE of Prophet: 0.20726886172177675 compared with: 0.22116804495195785 for original MLCSU projections
```

This December and January atypical



Intended next steps

- Use full dataset (2009 onwards)
- Include more ‘holidays’, including Tuesdays after Bank Holidays
- School holiday dummies-ideally generalise to not need local lists
- Check residuals
- Try methods of dealing with Covid period
- Cross-validation, examine performance for different periods
- Robustly compare performance with other approaches
- Rewrite in R!
- Productionise to automate forecast production
- Compare performance across many sites and measures

- Incorporate effects of weather-working with Lancaster University