Predicting Visitors to Edinburgh and Craigmillar Castles

Dr Caterina Constantinescu

May 24, 2019

Outline

1. Aims, data description & visualisation
2. Model types
3. GAMM simple model and forecast
4. External predictor data
   - Data collection and pre-processing
   - 'Concurrent' vs. lagged predictors
5. Model selection
   - 'Best' model
   - More on finding the 'best' model...
6. Caveats & extras
   - Assumptions
   - Going further

About me... My background

Data scientist @ The Data Lab

caterina.constatinescu@thedatalab.com

http://datapowered.io/

CaterinaC

- Romania
- BSc Psychology
- MSc Psychological Research → switch to R
- PhD Psychology

- Research background: `emotion`, `methods research`, `VR`
- *Are there differences between methods of eliciting emotions in the lab, and are any of these a good repr. of 'real-life' emotions?* (→ Android app)
- EdinbR organiser: @edinb_r EdinburghRusers
- DataTech organiser: https://www.datafest.global/data-tech

Outline

Data description & visualisation

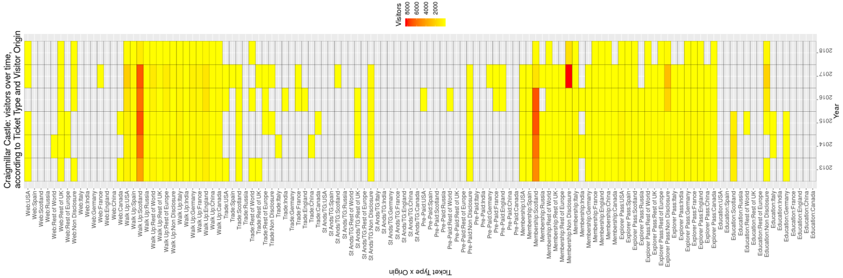
- **Aim**: Explain visitor trends over time at Edinburgh and Craigmillar Castles
  - Hence, collect ‘plausible’ predictor data for this purpose...
- **Data**: Time series available in daily & monthly format:
  - Split by: Visitors’ country of origin + Ticket Type purchased for the visit
  - Interval start: 2012 (Edinburgh Castle) or 2013 (Craigmillar Castle)
  - Interval end: March 2018

Data description & visualisation

	Month	Year	Date	Quarter	Region	Country	Site	TicketType	Visitors
6233	April	2014	2014-04-01	2014.2	NA	USA	Craigmillar	Explorer Pass	48
6234	May	2014	2014-05-01	2014.2	NA	USA	Craigmillar	Explorer Pass	53
6235	June	2014	2014-06-01	2014.2	NA	USA	Craigmillar	Explorer Pass	96
6236	July	2014	2014-07-01	2014.3	NA	USA	Craigmillar	Explorer Pass	74
6237	August	2014	2014-08-01	2014.3	NA	USA	Craigmillar	Explorer Pass	87
6238	September	2014	2014-09-01	2014.3	NA	USA	Craigmillar	Explorer Pass	42
6239	October	2014	2014-10-01	2014.4	NA	USA	Craigmillar	Explorer Pass	14
6240	November	2014	2014-11-01	2014.4	NA	USA	Craigmillar	Explorer Pass	6
6241	December	2014	2014-12-01	2014.4	NA	USA	Craigmillar	Explorer Pass	6
6242	January	2015	2015-01-01	2015.1	NA	USA	Craigmillar	Explorer Pass	6
6243	February	2015	2015-02-01	2015.1	NA	USA	Craigmillar	Explorer Pass	6
6244	March	2015	2015-03-01	2015.1	NA	USA	Craigmillar	Explorer Pass	21
6245	April	2015	2015-04-01	2015.2	NA	USA	Craigmillar	Explorer Pass	40
6246	May	2015	2015-05-01	2015.2	NA	USA	Craigmillar	Explorer Pass	86
6247	June	2015	2015-06-01	2015.2	NA	USA	Craigmillar	Explorer Pass	66
6248	July	2015	2015-07-01	2015.3	NA	USA	Craigmillar	Explorer Pass	105
6249	August	2015	2015-08-01	2015.3	NA	USA	Craigmillar	Explorer Pass	39
6250	September	2015	2015-09-01	2015.3	NA	USA	Craigmillar	Explorer Pass	39
6251	October	2015	2015-10-01	2015.4	NA	USA	Craigmillar	Explorer Pass	12
6252	November	2015	2015-11-01	2015.4	NA	USA	Craigmillar	Explorer Pass	3
6253	December	2015	2015-12-01	2015.4	NA	USA	Craigmillar	Explorer Pass	NA

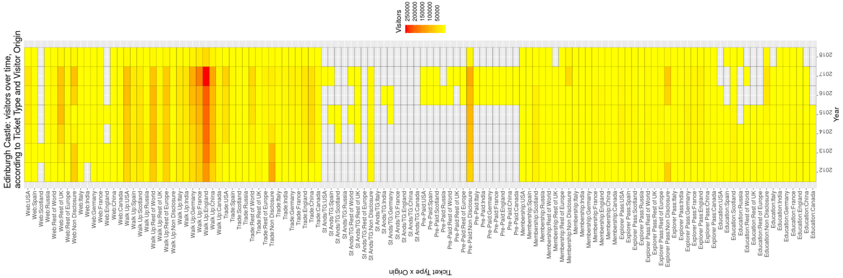
Data description & visualisation

Craigmillar Castle

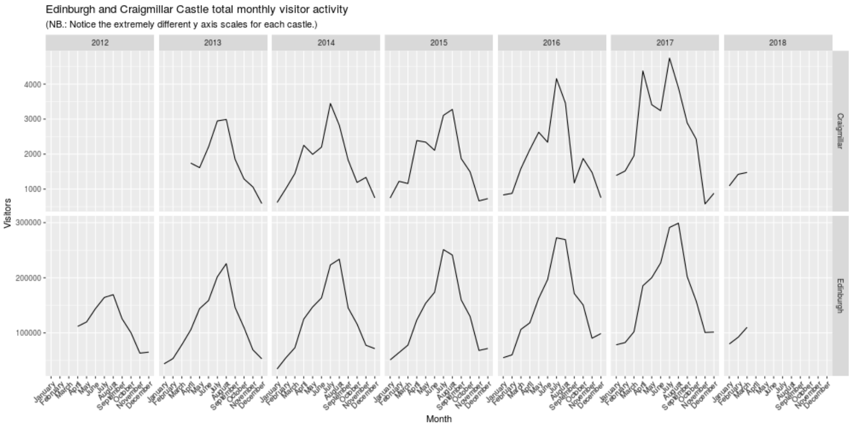


Data description & visualisation

Edinburgh Castle

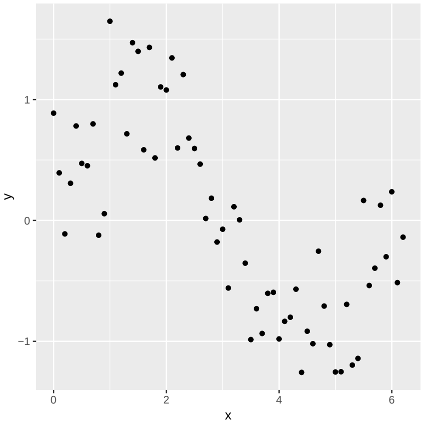


Data description & visualisation



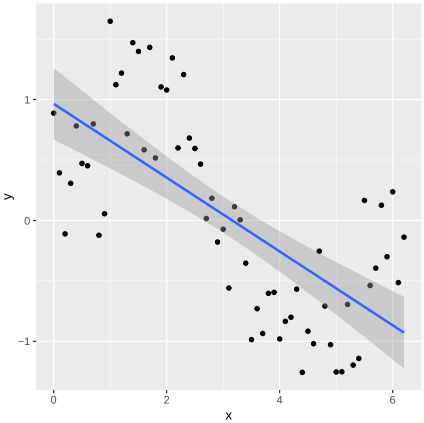
Outline

From LMs & GAMs...



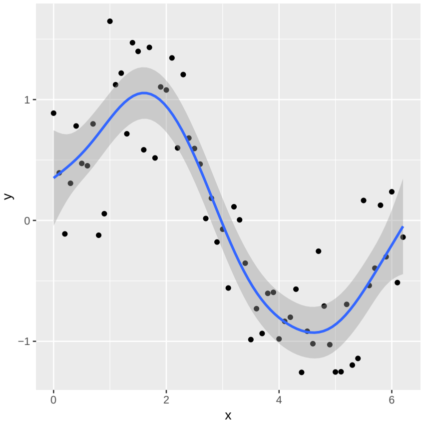
Data example courtesy of Mitchell Lyons, http://environmentalcomputing.net

From LMs & GAMs...



Data example courtesy of Mitchell Lyons, http://environmentalcomputing.net

From LMs & GAMs...



Data example courtesy of Mitchell Lyons, http://environmentalcomputing.net

From LMs & GAMs...

- **Linear** Models (not appropriate here):

$$y = b_0 + b_1 x_1 + b_2 x_2 + e$$

```
lm_mod <- lm( Visitors ~ TicketType + Site + Country, data = dat )
```

- Generalised Additive Models (GAMs):

$$y = b_0 + f_1(x_1) + f_2(x_2) + e$$

```
gam_mod <- gam( Visitors ~
                  s( Month ) +
                  s( as.numeric( Date ) ) +
                  te( Month, as.numeric( Date ) ) + # But see ?te and ?ti
                  Site,
                data = dat )
```

From LMs & GAMs, to MLMs & GAMs

Mixed Linear Models (MLMs):

```
mlm_mod <- lme( fixed = Visitors ~ TicketType + Site + Country + poly( Month, 2 ) + as.factor( Year ),
                random = ~ Site,
                data = dat,
                cor = corARMA( form = ~ 1 | Site,  p = 1, q = 0 )
)
```

Generalised Additive MIXED Models (GAMMs):

```
# mgcv::gamm() = nlme::lme() + mgcv::gam()

gamm_mod <- gamm( Visitors ~
                          s( Month ) +
                          s( as.numeric( Date ) ) +
                          te( Month, as.numeric( Date ) ) +
                          Site,
                        data = dat,
                        correlation = corARMA( p = 1, q = 0 )
                        # random = list( Site = ~ 1 )
                        )
```

More on GAM(M)s...

Splines = 'the essential building block of a GAM' (Noam Ross). They're built from lots of little functions, i.e., basis functions.

Types of splines:

- Thin plate
  - Default option in `mgcv`
  - "Choose how many basis functions are to be used and then solve the problem of finding the set of this many basis functions that will optimally approximate a full spline." (Simon Wood)
- Cubic regression
  - 'Modest'-sized set of knots spread evenly through the data
  - Fit a cubic polynomial between every pair of knots.
  - Have continuous first and second derivatives at each knot

More on GAM(M)s...

- Gaussian Process - incorporates autocorrelation
- Tensor product (smooth interactions between vars, scale-invariant)
- Find out more from `?mgcv::smooth.terms`

Other concepts:

- GCV = Estimate prediction error
  - "I suppose the ideal value might be 0 (or close to it) as it is an estimator of the mean square error. Because it uses deviations from the observed data, its value depends on the values of the response. So treat it like AIC in the sense that smaller is better." (Gavin Simpson, Stats SE, Apr 23 '18)
  - In mgcv::gam(), but not gamm()
- EDF = 'wiggliness' → Penalisation: fit the data, not the noise.
- Adj R^2 (in `gamm()`)

Check out:
`?mgcv::gam.models`
http://environmentalcomputing.net/intro-to-gams/

Outline

GAMM simple model and forecast

Steps:

- Aggregate the data (collapse countries and ticket types, but keep sites separate ofc)
- Predict the data from itself:
  - spline for **Month**, **Time** (Date), and their **interaction**
  - all three split by **Site**
  - **autoregressive** term (lag 1)

GAMM simple model and forecast

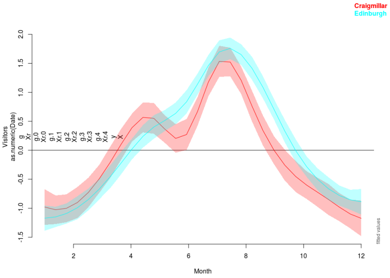
```r
library( data.table )
library( itsadug )
library( mgcv )
library( ggplot2 )

simple_GAMs_data <- aggregate( Visitors ~ Month + Year + Date + Site,
                               data = dat,
                               FUN = sum )
setDT( simple_GAMs_data )

# Standardise scores as Edin castle is vastly more popular than Craigmillar:
simple_GAMs_data[ , Visitors := ave( Visitors, Site, FUN = scale ) ]

# Model:
gamm_few_preds <- gamm( Visitors ~
                          s( Month, bs = "cc", by = Site ) +
                          s( as.numeric( Date ), by = Site ) +
                          te( Month, as.numeric( Date ), by = Site ) +
                          Site,
                        data = simple_GAMs_data,
                        control = lmeControl( opt = "optim", msMaxIter = 10000 ),
                        correlation = corARMA( p = 1, q = 0 )
                        # random = list( Site = ~ 1 ) # Random intercepts by site
                        # correlation = corAR1( ) )
                        )
```

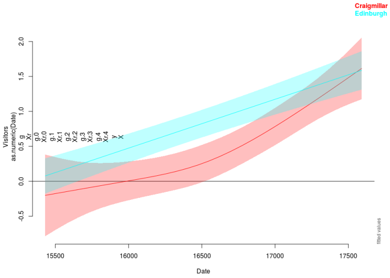
GAMM simple model and forecast

```
# Using $gam side:

> summary( gamm_few_preds$gam )

Family: gaussian
Link function: identity

Formula:
Visitors ~ s(Month, bs = "cc", by = Site) + s(as.numeric(Date),
    by = Site) + te(Month, as.numeric(Date), by = Site) + Site

Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -0.04312    0.03878  -1.112   0.2687
SiteEdinburgh  0.08738    0.05119   1.707   0.0908 .
---
Signif. codes:  0   ***   0.001   **   0.01   *   0.05   .   0.1       1

Approximate significance of smooth terms:
                                             edf Ref.df      F              p-value
s(Month):SiteCraigmillar                   7.274  8.000 22.707 < 0.0000000000000002 ***
s(Month):SiteEdinburgh                     6.409  8.000 63.818 < 0.0000000000000002 ***
s(as.numeric(Date)):SiteCraigmillar        1.001  1.001 38.212     0.00000000931069 ***
s(as.numeric(Date)):SiteEdinburgh          1.000  1.000 57.220     0.00000000000612 ***
te(Month,as.numeric(Date)):SiteCraigmillar 6.276  6.276  2.935              0.00787 **
te(Month,as.numeric(Date)):SiteEdinburgh   3.212  3.212  3.316              0.02018 *
---
Signif. codes:  0   ***   0.001   **   0.01   *   0.05   .   0.1       1

R-sq.(adj) =  0.913
  Scale est. = 0.081956   n = 132
```

GAMM simple model and forecast

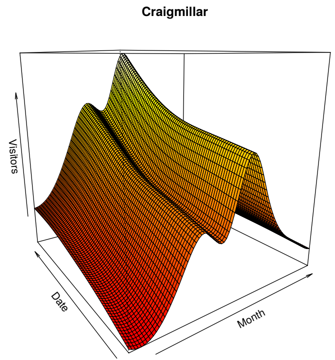
```
# Using $gam side:

itsadug::plot_smooth( gamm_few_preds$gam, view = "Month", plot_all = "Site", rug = F )
itsadug::plot_smooth( gamm_few_preds$gam, view = "Date", plot_all = "Site", rug = F )
```

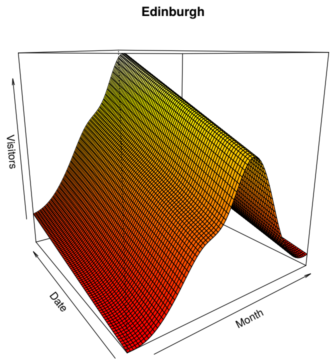




GAMM simple model and forecast

```
# Using $gam side:

mgcv::vis.gam( gamm_few_preds$gam,
               view = c( "Month", "Date" ),
               zlab = "Visitors", cond = list( Site = "Craigmillar" ),
               n.grid = 80, theta = 325,
               plot.type = "persp", main = "Craigmillar", type = "response" )
```





GAMM simple model and forecast

```
# Creating a forecast

future_dates <- seq( as.Date( "2018-04-01" ), as.Date( "2020-04-01" ), by = "month" )
future_sites <- c( rep( "Edinburgh", length( future_dates ) ),
                   rep( "Craigmillar", length( future_dates ) ) )
future_dates <- rep( future_dates, 2 )
new_data <- data.table( Date = future_dates,
                        Site = future_sites )
new_data[ , Month := as.numeric( format( Date, "%m" ) ) ]
new_data[ , Year := as.numeric( format( Date, "%Y" ) ) ]

new_data <- rbind( simple_GAMs_data, new_data, fill = TRUE )
setnames( new_data, "Visitors", "Observed" )

# Get fitted values PLUS forecast over next 2 years all in one go:
predictions <- predict( gamm_few_preds$gam, newdata = new_data, se.fit = TRUE )
new_data[ , Predicted := predictions[[1]] ]
new_data[ , SE := predictions[[2]] ]

new_data_long <- melt( new_data,
                       id.vars = c( "Month", "Year", "Date", "Site", "SE" ),
                       measure.vars = c( "Observed", "Predicted" ),
                       variable.name = "Visitors",
                       value.name = "Value" )
new_data_long[ , SE := ifelse( Visitors == "Observed", NA, SE ) ]

new_data_long[ , CILower := Value - ( 1.96 * SE ) ]
new_data_long[ , CIUpper := Value + ( 1.96 * SE ) ]

# Alternative: itsadug::get_predictions()
```

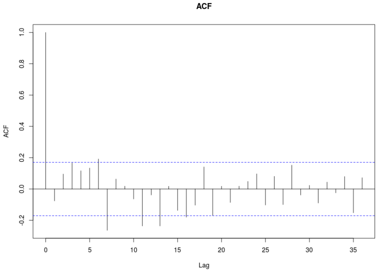
GAMM simple model and forecast

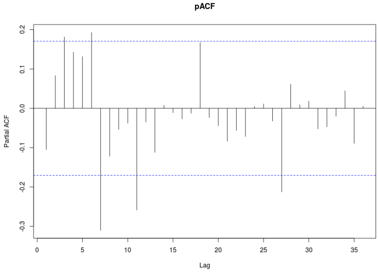
```
# Now get the forecast info and plot using ggplot2:

ggplot( data = new_data_long,
        aes( y = Value, x = Date, color = Visitors ) ) +
  geom_ribbon( aes( ymin = CILower,
                    ymax = CIUpper ),
               alpha = 0.3, linetype = 0 ) +
  geom_line( lwd = 1.15, alpha = 0.6 ) +
  scale_color_manual( values = c( "black", "red" ) ) +
  facet_wrap( ~ Site ) + #, scales = "free"
  labs( x = "Date", y = "Visitors (scaled)" ) +
  scale_x_date( date_breaks = "2 months" ) +
  theme( axis.text.x = element_text( angle = 90,
                                     hjust = 1 ),
         text = element_text( size = 15.5 ) ) +
  ggtitle( "Predicted vs. observed visitor numbers across both castles" )
```

GAMM simple model and forecast

GAMM simple model and forecast

```
# Using $lme side:

acf( resid( gamm_few_preds$lme, type = "normalized" ), lag.max = 36, main = "ACF" )
pacf( resid( gamm_few_preds$lme ), lag.max = 36, main = "pACF" )
```





Outline

Data collection, merges, dimension reduction

- The Consumer Confidence Index (CCI):

  > *[...] provides an indication of [...] households' [...] expected financial situation, their sentiment about the general economic situation, unemployment and capability of savings.*

- Google Trends: searches for 'Edinburgh Castle' and 'Craigmillar Castle', by country, over time.
- For-Sight hotel bookings: 4 Edinburgh-based hotels (no. of nights, no. of adults, date etc.)
- GDELT world event data (no. of mentions, articles, tone etc. covering events in Scotland):

  > *[...] monitors the world's news media from nearly every corner of every country in print, broadcast, and web formats, in over 100 languages*

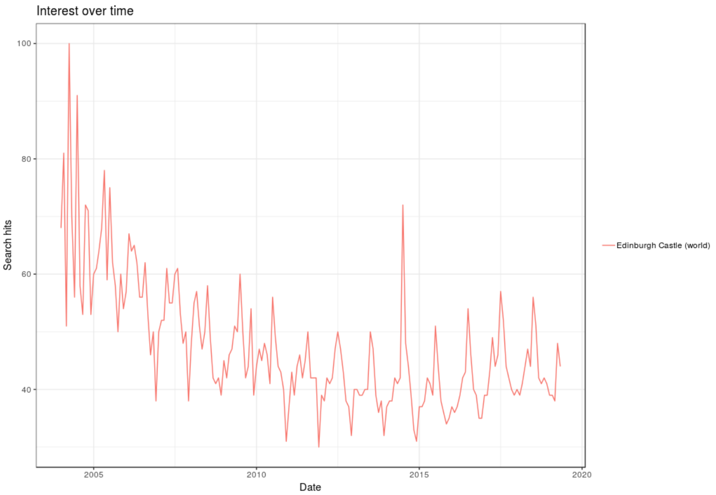
- IMDb & Open Movie Database (OMDb): no. of votes, average rating etc. for films related to Scotland.
- Local weather (temperature, amount of precipitation etc.)
- Skyscanner flight searches

Data collection, merges, dimension reduction

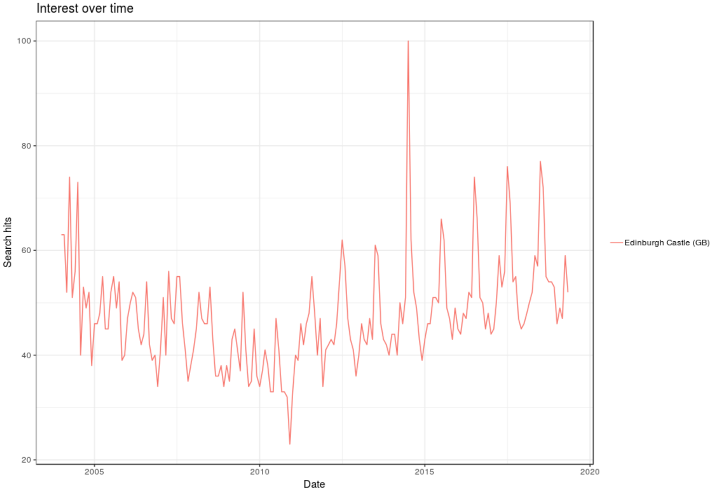
```
library( gtrendsR )

edinburgh_castle <- gtrends( keyword = 'Edinburgh Castle',
                             geo = '',
                             time = 'all' )
```

- time = 'all' - Since the beginning of Google Trends (2004)
- geo = across Globe unless you specify a particular country code

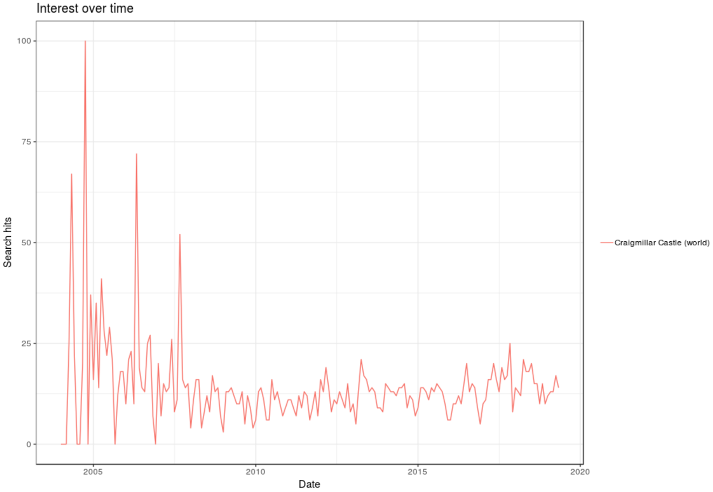
Data collection, merges, dimension reduction



Data collection, merges, dimension reduction



Data collection, merges, dimension reduction



GAMM: 'Concurrent' vs. lagged predictors

'Concurrent' vs. lagged...

Approach:

- Create a model including all the 'concurrent' predictors and refine it as much as possible:
  - Splines for Month, Date, their interaction (all split by Site)
  - Main effects for Site and Ticket Type (Country ns.)
  - Random effects to capture any variations across site x country x ticket type combinations
  - Autoregressive correlations
  - **Number of Adults** (For-Sight hotel data) & **Temperature**
- What about **CCI**, **Number of articles** (GDELT), **IMDb votes**, and Google Trends **hits**?

GAMM: 'Concurrent' vs. lagged predictors

- All possible permutations ($N = 2{,}401$) for lag amounts of 0 mths to 6 mths, e.g.,

	CCI	NumArticles	IMDbVotes	hits
1758	5	0	6	0
1470	4	1	6	6
1749	5	0	4	5
35	0	0	4	6
460	1	2	2	4

- Created a lagged dataset for each of these
- Specified a model for each, including all previous terms, as well as the lagged terms
- Chose the best-fitting model from all 2,401 based on R^2.

Top models

1. CCI: 6 ; NumArticles:1 ; imdbVotes: 4 ; hits: 4
2. CCI: 4 ; NumArticles:1 ; imdbVotes: 4 ; hits: 6
3. CCI: 6 ; NumArticles:5 ; imdbVotes: 4 ; hits: 4
4. CCI: 5 ; NumArticles:1 ; imdbVotes: 4 ; hits: 6
5. CCI: 6 ; NumArticles:1 ; imdbVotes: 4 ; hits: 6
6. CCI: 3 ; NumArticles:1 ; imdbVotes: 4 ; hits: 6
7. CCI: 6 ; NumArticles:6 ; imdbVotes: 4 ; hits: 4
8. CCI: 6 ; NumArticles:4 ; imdbVotes: 4 ; hits: 4
9. CCI: 6 ; NumArticles:3 ; imdbVotes: 4 ; hits: 4
10. CCI: 4 ; NumArticles:5 ; imdbVotes: 4 ; hits: 6

R^2 varies between 0.46 and 0.56 across all 2,401 models.

But can we improve this further...?

Outline

Refining the 'best' model from previous step

```
refining_best_gamm <-
  gamm( Visitors ~
          s( Month, bs = "cc", by = Site )  +
          s( as.numeric( Date ), by = TicketType, bs = "gp" ) +
          te( Month, as.numeric( Date ), by = Site ) +
          s( NumberOfAdults ) +
          s( Temperature ) +

          Site +
          TicketType +

          s( LaggedCCI ) +
          s( LaggedNumArticles ) +
          s( LaggedimdbVotes ) +
          s( Laggedhits, by = Site ),

        data = best_lag_solution,

        control = lmeControl( opt = "optim", msMaxIter = 10000 ),
        random = list( GroupingFactor = ~ 1 ),
        # GroupingFactor = Site x TicketType x Country
        REML = TRUE,
        correlation = corARMA( p = 1, q = 0 )
  )
```

Now every ticket type (rather than site) can evolve differently over time
$\Rightarrow$ *adj.* $R^2 = 0.694$

Refining the 'best' model from previous step

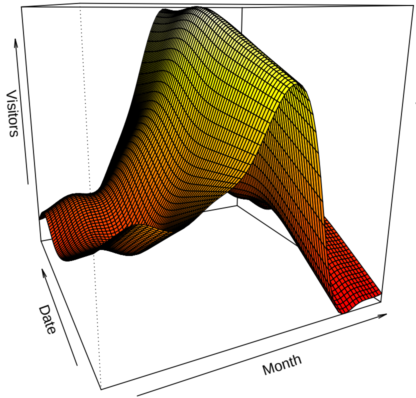
```
Family: gaussian
Link function: identity

Formula:
Visitors ~ s(Month, bs = "cc", by = Site) + s(as.numeric(Date),
    by = TicketType, bs = "gp") + te(Month, as.numeric(Date),
    by = Site) + s(NumberOfAdults) + s(Temperature) + Site +
    TicketType + s(LaggedCCI) + s(LaggedNumArticles) + s(LaggedimdbVotes) +
    s(Laggedhits, by = Site)

Parametric coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.06296    0.06656   0.946 0.344439
SiteEdinburgh            0.22692    0.06350   3.573 0.000372 ***
TicketTypeEducation      0.07420    0.14545   0.510 0.610085
TicketTypeExplorer Pass -0.22165    0.06590  -3.363 0.000804 ***
TicketTypeMembership    -0.43625    0.06684  -6.527 1.14e-10 ***
TicketTypePre-Paid      -0.93159    0.14608  -6.377 2.91e-10 ***
TicketTypeTrade         -0.09395    0.06916  -1.358 0.174698
TicketTypeWeb           -0.10447    0.06915  -1.511 0.131229
---
Signif. codes:  0    ***    0.001    **    0.01    *    0.05    .    0.1         1
```

Refining the 'best' model from previous step

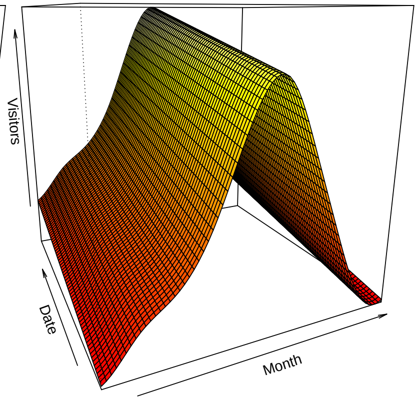
```
Formula:
Visitors ~ s(Month, bs = "cc", by = Site) + s(as.numeric(Date),
    by = TicketType, bs = "gp") + te(Month, as.numeric(Date),
    by = Site) + s(NumberOfAdults) + s(Temperature) + Site +
    TicketType + s(LaggedCCI) + s(LaggedNumArticles) + s(LaggedimdbVotes) +
    s(Laggedhits, by = Site)

Approximate significance of smooth terms:
                                                edf Ref.df      F  p-value
s(Month):SiteCraigmillar                   4.954928  8.000  7.941 6.94e-15 ***
s(Month):SiteEdinburgh                     7.281621  8.000 20.031  < 2e-16 ***
s(as.numeric(Date)):TicketTypeWalk Up      1.000001  1.000  2.854  0.09149 .
s(as.numeric(Date)):TicketTypeEducation    1.000002  1.000  1.658  0.19821
s(as.numeric(Date)):TicketTypeExplorer Pass 1.000014  1.000  2.690  0.10132
s(as.numeric(Date)):TicketTypeMembership   7.325579  7.326 27.817  < 2e-16 ***
s(as.numeric(Date)):TicketTypePre-Paid     1.000004  1.000  0.231  0.63072
s(as.numeric(Date)):TicketTypeTrade        1.000000  1.000  6.328  0.01206 *
s(as.numeric(Date)):TicketTypeWeb          1.000000  1.000 22.546 2.39e-06 ***
te(Month,as.numeric(Date)):SiteCraigmillar 0.001085 15.000  0.000  0.31354
te(Month,as.numeric(Date)):SiteEdinburgh   5.443975 15.000  2.090 4.26e-07 ***
s(NumberOfAdults)                          1.000009  1.000 40.617 2.93e-10 ***
s(Temperature)                             5.675452  5.675  3.058  0.00436 **
s(LaggedCCI)                               1.000019  1.000  3.921  0.04798 *
s(LaggedNumArticles)                       1.985548  1.986  5.661  0.00598 **
s(LaggedimdbVotes)                         2.702820  2.703  5.187  0.00188 **
s(Laggedhits):SiteCraigmillar              1.000011  1.000  8.012  0.00475 **
s(Laggedhits):SiteEdinburgh                3.106882  3.107  3.756  0.01092 *
---
Signif. codes:  0  ***  0.001  **  0.01  *  0.05  .  0.1      1

R-sq.(adj) =  0.694   Scale est. = 0.3072    n = 932
```

Month x Date

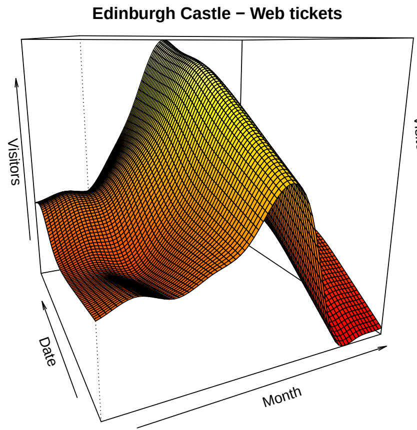
Edinburgh Castle − Walk Up tickets

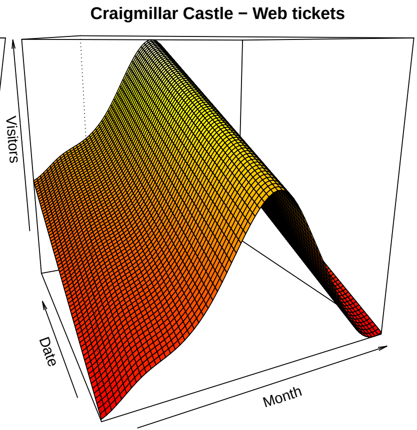


Craigmillar Castle − Walk Up tickets



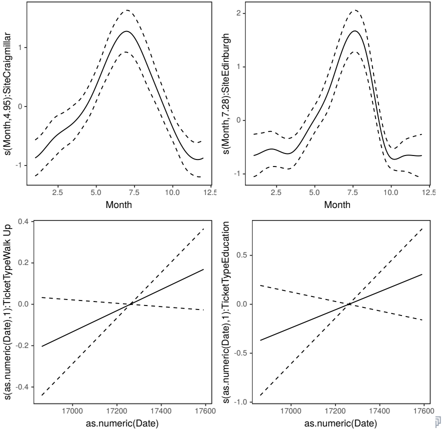
Month x Date


Edinburgh Castle − Web tickets
Visitors
Date
Month

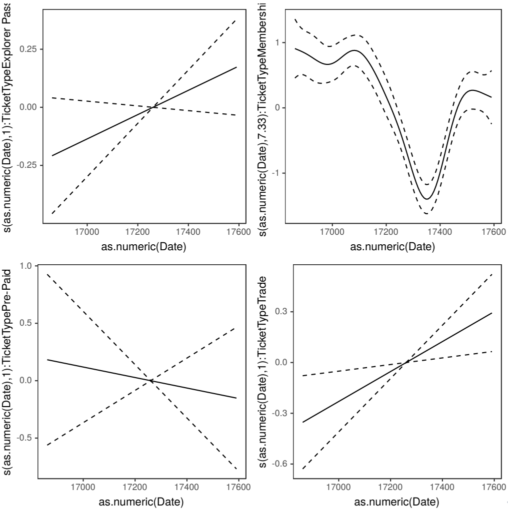

Craigmillar Castle − Web tickets
Visitors
Date
Month

Individual smooths

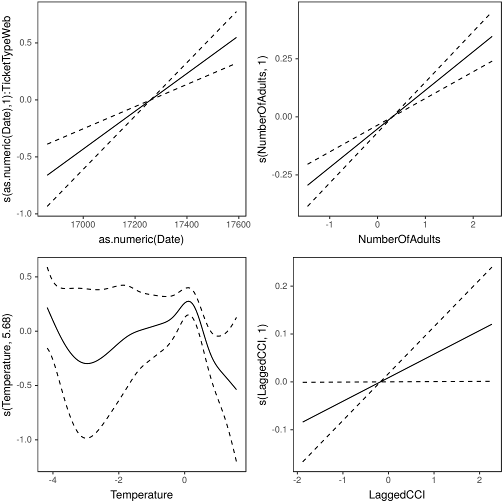
```
g <- mgcViz::getViz( refining_best_gamm$gam )
print( plot( g, allTerms = TRUE ), pages = 1 )
print( plot( g, select = c( 1:9, 12:20 ) ), pages = 5, seWithMean = TRUE )
```



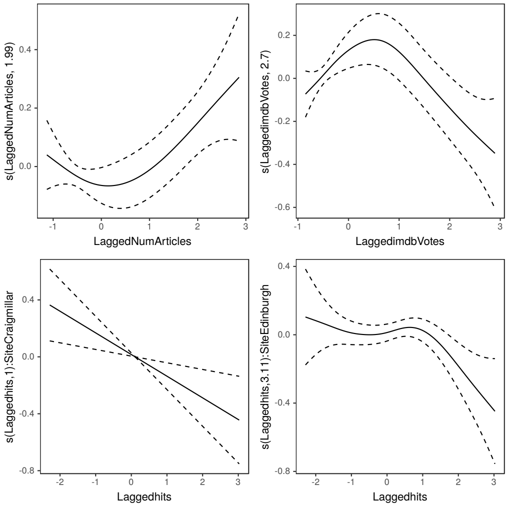
Individual smooths



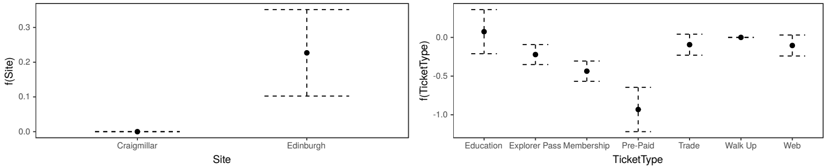
Individual smooths



Individual smooths



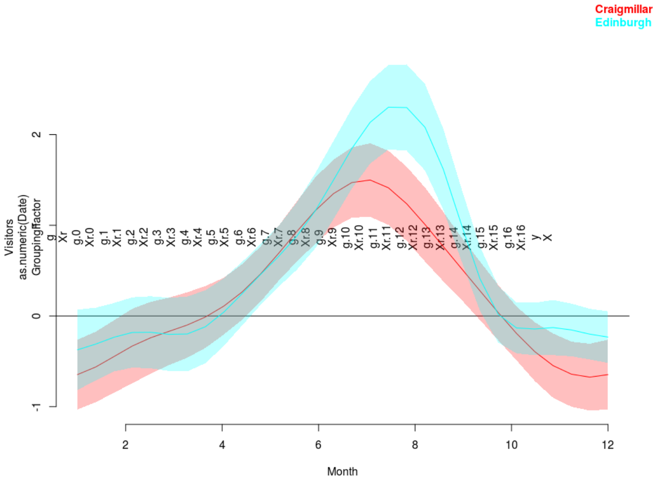
Factors


f(Site)
0.3
0.2
0.1
0.0
Craigmillar
Edinburgh
Site
f(TicketType)
0.0
-0.5
-1.0
Education
Explorer Pass
Membership
Pre-Paid
Trade
Walk Up
Web
TicketType

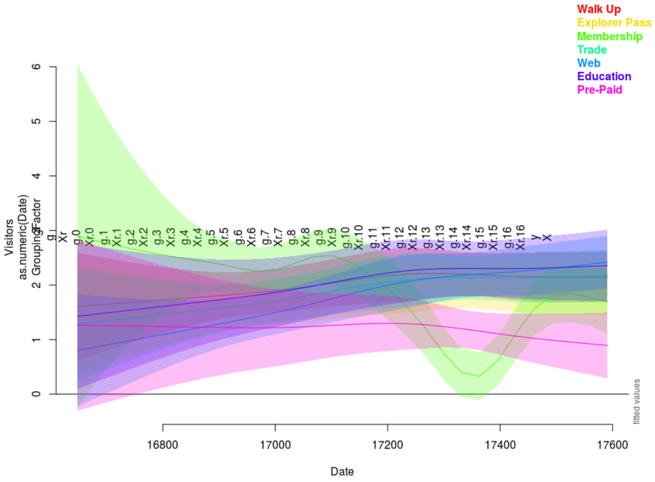
Individual smooths

```
> itsadug::plot_smooth( refining_best_gamm$gam,
                        view = "Month",
                        plot_all = "Site",
                        rug = F, hide.label = T )



Summary:
        * Site : factor; set to the value(s): Craigmillar, Edinburgh.
        * TicketType : factor; set to the value(s): Walk Up.
        * Month : numeric predictor; with 30 values ranging from 1.000000 to 12.000000.
        * Date : numeric predictor; set to the value(s): 17136.
        * NumberOfAdults : numeric predictor; set to the value(s): -0.114300092534047.
        * Temperature : numeric predictor; set to the value(s): 0.169091887898448.
        * LaggedCCI : numeric predictor; set to the value(s): -0.527443903017371.
        * LaggedNumArticles : numeric predictor; set to the value(s): -0.343637299456009.
        * LaggedimdbVotes : numeric predictor; set to the value(s): -0.275229099568131.
        * Laggedhits : numeric predictor; set to the value(s): -0.00755797998814945.
```

Individual smooths



Individual smooths



More on finding the 'best' model...

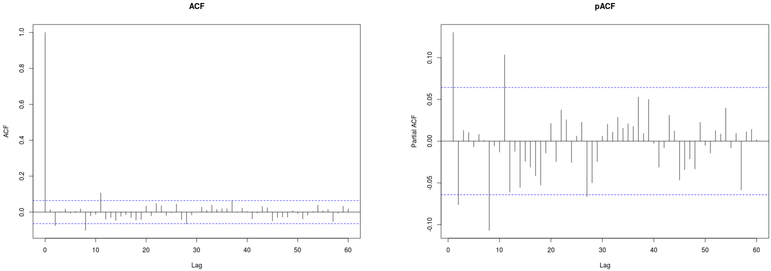
- `mgcv::gam()` options:
  - `?anova.gam`
  - 'Adding a penalty on the null space of each smooth', with the `select = TRUE` argument. (Simon Wood, More advanced use of `mgcv`: https://people.maths.bris.ac.uk/~sw15190/mgcv/tampere/mgcv-advanced.pdf)
  - `?AIC`
  - Backward selection via GCV
- Other GAMM options:
  - `gamm4::gamm4()`, then `MuMIn::dredge()`

The fine print for `mgcv::gamm()`:

"GAMM uses penalized quasi-likelihood (PQL). According to some statisticians (including Brian Ripley, who wrote the original PQL code in MASS::glmmPQL, which might be what GAMM relies on – I don't remember, it might incorporate its own PQL code), one shouldn't use likelihood-based approaches (including AIC) with PQL algorithms, because they don't estimate a true likelihood".
(https://stat.ethz.ch/pipermail/r-sig-mixed-models/2013q1/020055.html)

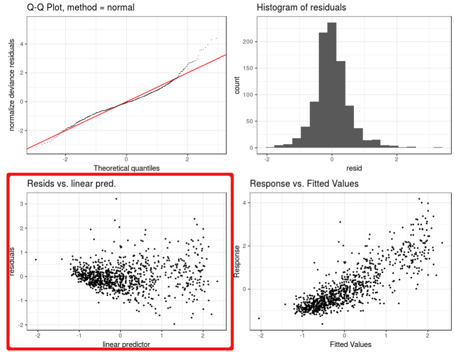
Outline

Assumptions



Assumptions

Assumptions: `??mgcViz; ?mgcViz::check.gamViz`



From `?gam.check`: "Take care when interpreting results from applying this function to a model fitted using gamm. In this case the returned gam object is based on the working model used for estimation, and will treat all the random effects as part of the error. This means that the residuals extracted from the gam object are not standardized for the family used or for the random effects or correlation structure. Usually it is necessary to produce your own residual checks based on consideration of the model structure you have used."

Going further

- Mitchell Lyons' post:
  http://environmentalcomputing.net/intro-to-gams/
- Gavin Simpson's blog:
  https://www.fromthebottomoftheheap.net/blog/
- Noam Ross - Nonlinear Models in R: The Wonderful World of mgcv:
  https://www.youtube.com/watch?v=q4_t8jXcQgc
- Josef Fruehwald - Studying Pronunciation Changes with gamms:
  http://edinbr.org/edinbr/2017/10/10/october-meeting.html

Hopefully not too boring...

Any questions?