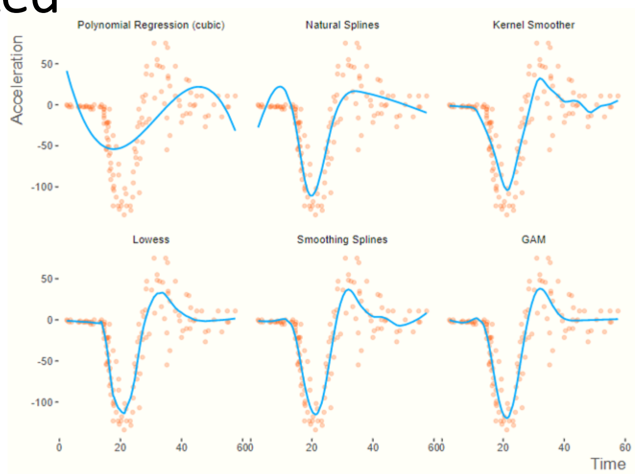
NAIC NATIONAL ASSOCIATION OF INSURANCE COMMISSIONERS

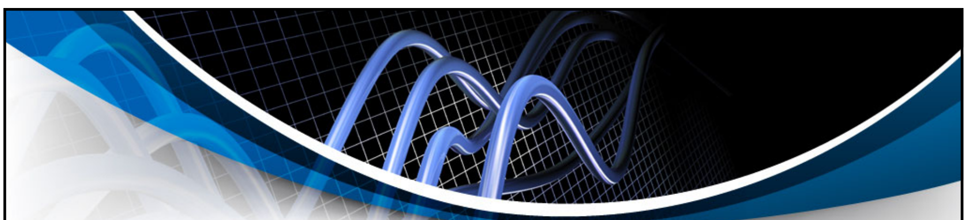
2021 NAIC Insurance Summit

Generalized Additive Models (GAMs)

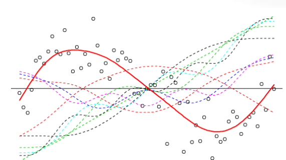
Dorothy L. Andrews, FCA, ASA, MAAA, CSPA



1



Agenda



- GLM vs. GAM?
- Theoretical Form of a GAM
- Basis Functions Defining GAMs
- Model Results of a GAM by Example
- Concurvity Concerns in Non-Linear Models
- GAM References

2

GLM vs. GAM?

Generalized Linear Model (GLM)

$$g(\mu) = X\beta$$

$$g(\mu) = \eta$$

$$E(y) = \mu = g^{-1}(\eta)$$

$$g(\mu) = b_0 + \mathbf{b_1}X_1 + \mathbf{b_2}X_2 \ldots + \mathbf{b_p}X_p$$

Generalized Additive Model (GAM)

$$y \sim \text{Exponential Family}(\mu, \text{etc.})$$

$$\mu = E(y)$$

$$g(\mu) = b_0 + \mathbf{f}(x_1) + \mathbf{f}(x_2) + \ldots + \mathbf{f}(x_p)$$

g() is the link function

Key Difference: GLMs add constant factors of variables.
GAMs add functions of variables.

Linear Basis Model

If the relationship between the inputs and the target is non-linear, we use linear basis function models to express relationship.

These models assume that the target is a linear combination of a set of p+1 basis functions.

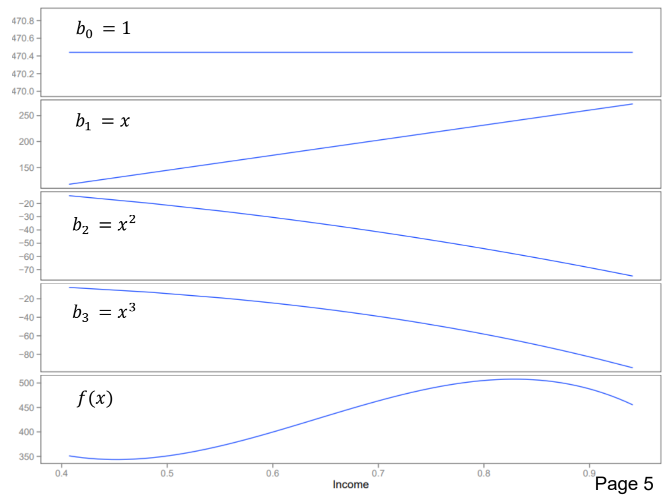
$$Y_i = \beta_0 + \beta_1 \emptyset_1(x_1) + \beta_2 \emptyset_2(x_2) + \ldots + \beta_p \emptyset_p(x_p)$$

A basis is a set of basis functions $\emptyset_j$ that will be combined to produce *f(x)*:

$$f(x) = \sum_{j=1}^{p} \beta_j \emptyset_j(x)$$

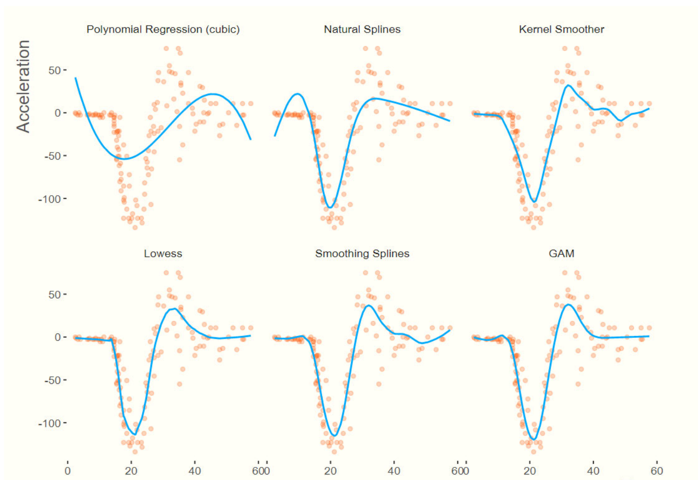
Polynomial Basis Example

$$f(x) = \beta_0 1 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$



Motorcycle Accident Data

Simple transformations don't work well here!



Measurements of head acceleration/deacceleration in a simulated motorcycle accident after impact, used to test crash helmets.

A GAM Basis

A GAM is a sum of *smooth functions* or *smooths*

$$Y_i = \beta_0 + \sum_{j=1}^{p} s_j(x) + \epsilon_i$$

$$\text{where } \emptyset_j(x) = s_j(x)$$
$$\epsilon_i = \text{error term}$$

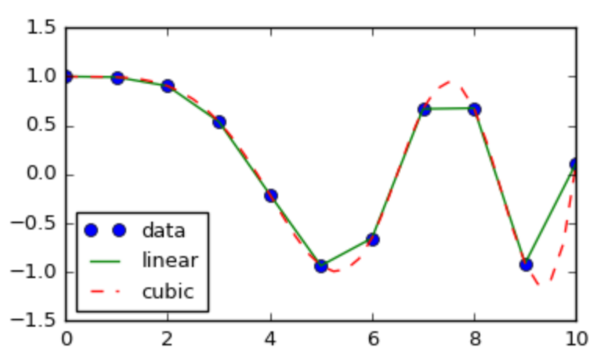
Note: There are many smooth functions we can use as basis functions

In R, we use

- *library (mgcv) #The mgcv package*
- *gam() #The glm() equivalent for GAMs*

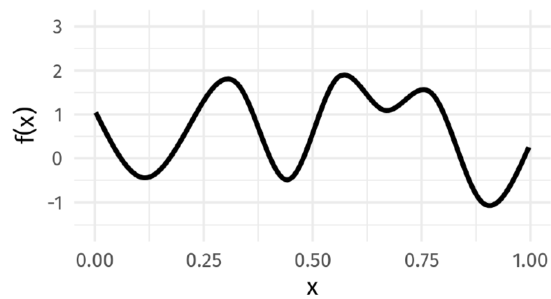
Note: *mgcv* stands for mixed GAM computational vehicle

Mathematical Splines



Cubic Splines Interpolation is piecewise interpolation with a different cubic equation between each pair of data points. These points are also called "knots." Cubic interpolation creates a smooth fit at the knots.

Wiggly Functions: Splines



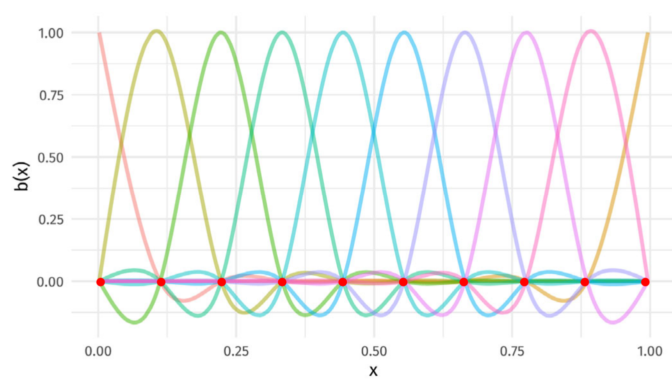
Splines are *functions* composed of simpler functions

$$s(x) = \sum_{k=1}^{K} \beta_k b_k(x)$$

Resultant spline is a sum of weighted basis functions, evaluated at the values of x.

K = number of basis functions.

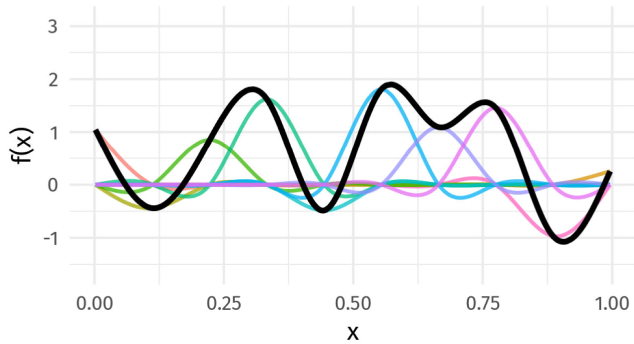
Spline formed from basis functions



- These are the “knots.” They are the boundaries of the piecewise splines that define the GAM.

GAM Model

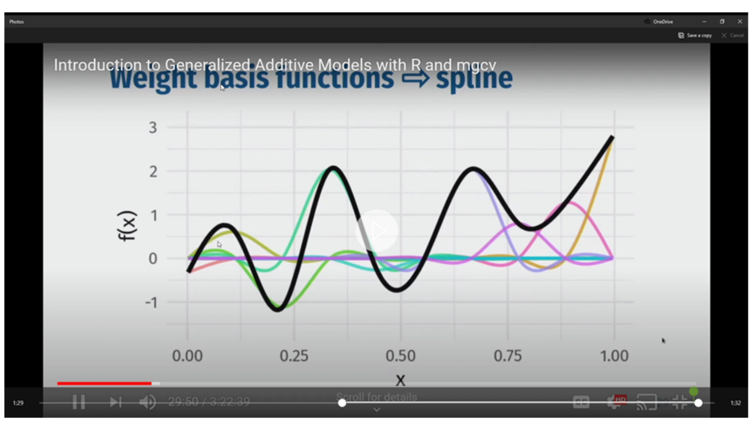
Weight basis functions ⇨ spline



Page 11

11

GAM Model Fitting

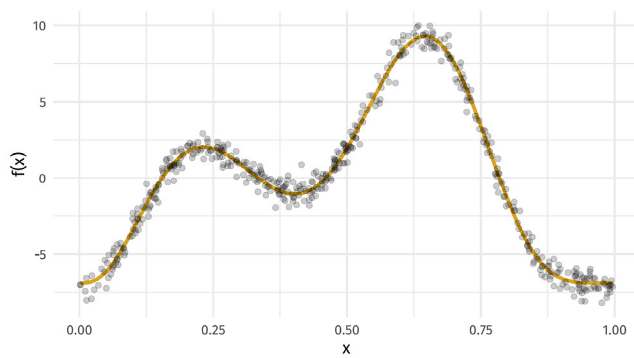


The GAM spline is the sum of all the underlying basis functions.

Page 12

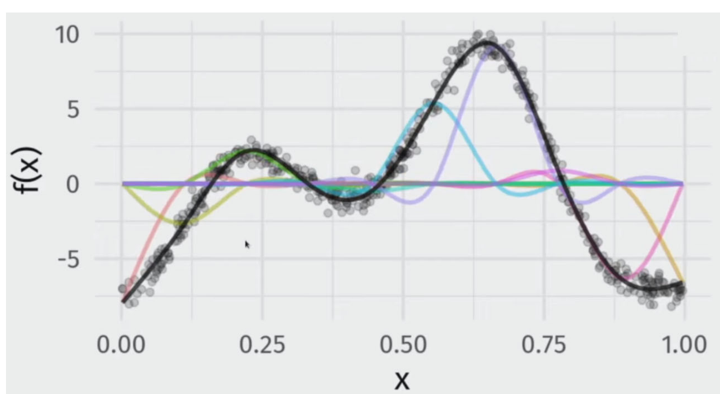
12

Estimating GAM Betas



GAMs work well fitting wiggly data because there is no single polynomial to fit this data.

Estimating GAM Betas



You can see how the individual basis functions are summed to estimate the coefficients for each basis function.

Penalized Log-Likelihood

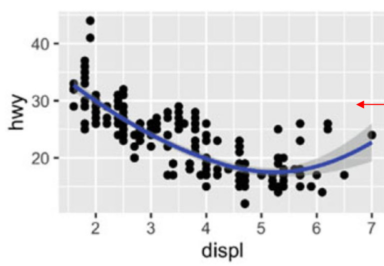
$$L_p = \underbrace{L(\beta)} - \underbrace{\frac{1}{2}\lambda\beta^T S\beta}$$

Maximum Likelihood as in the GLM

Penalty to discourage overfitting - wiggliness

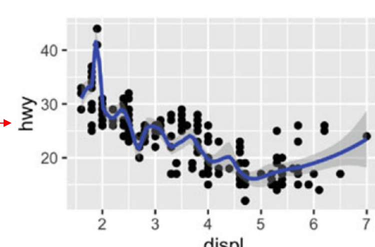
The more "wiggly" the fit the more the model overfits and the greater the penalty. The smoothing parameter λ controls how much penalty is paid for the wiggliness of the model. It balances the fit of the data with the wiggliness or complexity of the model.

Penalized Log-Likelihood



A Not Very Wiggly Fit
Small Penalty

A Wiggly Fit
Bigger Penalty



Wiggliness Penalty

$$\int_R [f'']^2 dx = \beta^T S \beta = W$$

- The LHS represents the curvature or the rate of change in the slope which is the 2nd derivative.
- The second derivative is squared so that concave and convex sections of the curve, which intuitively should both contribute equally to "wiggliness" if they are the same shape, both contribute equally to "wiggliness" if they are the same shape.
- The integral can be written as $\beta^T S \beta$, where S is a penalty matrix created from basis functions.
- W stands for “wiggliness.” Zero wiggliness = Straight line

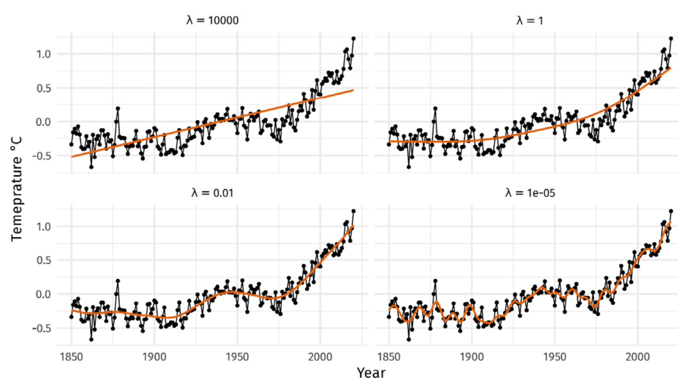
Page 17

17

The Effect of λ

HadCRUT4 is a global temperature dataset, providing gridded temperature anomalies across the world as well as averages for the hemispheres and the globe as a whole.

HadCRUT4 time series



The smaller the λ the wigglier the fit.

Page 18

18

Estimating λ

There are two approaches:

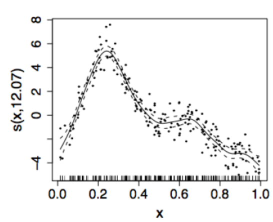
1. Predictive: Minimize out-of-sample error
   - AIC
   - Mallow's C_p
   - GCV (Generalized Cross-Validation)

2. Bayesian: Put priors on our basis coefficients
   - REML (Restricted Maximum Likelihood) produces an unbiased ML estimator of the variance.
   - REML is numerically stable
   - R Function: gam(…, method = REML)

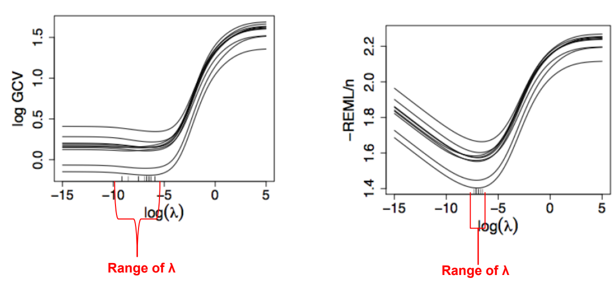
Zhang, X. (2015) A tutorial on restricted maximum likelihood estimation in linear regression and linear mixed-effects model. Retrieved from https://people.csail.mit.edu/xiuming/docs/tutorials/reml.pdf

GCV vs. REML

Data we to fit a Gam to →



Notice REML finds a λ in a smaller range than does GCV!



Wood, S. (2017). Generalized Additive Models: An Introduction with R, Second Edition.

Maximum Wiggliness

We set **basis complexity** or "size"

This is *maximum wiggliness*, can be thought of as number of small functions that make up a curve

Once smoothing is applied, curves have fewer **effective degrees of freedom (EDF)**

EDF < k

The penalty function works to reduce some basis coefficients to zero which reduces the *Degrees of Freedom (DF)* to *Effective Degrees of Freedom(EDF).*

This similar to Regularization Penalties.

Maximum Wiggliness

k must be *large enough*, the penalty does the rest

Large enough — space of functions representable by the basis includes the true function or a close approximation to the true function

Bigger k increases computational cost but need to make sure your smooths are **wiggly enough** to capture behavior of your data.

In **mgcv** (written by Simon Wood), default values are arbitrary — after choosing the model terms, this is the key user choice.

The software chooses λ.

Must be checked! — gam.check() – Will help assess goodness of k.

GAM Function in R

```
gam(formula,
        family=gaussian(), #Y ~ Independent Variables
        data=list(), #Model Data
        weights=NULL, #Data weights
        subset=NULL, #Optional Observations
        na.action, #How to handle NAs
        offset=NULL, #Model offset
        method="GCV.Cp", #Method to Estimate Smoothing Parameter
        optimizer=c("outer","newton"), #For Smoothing Parameter
        control=list(), #Control Variables
        scale=0, #Indicates Scale Parameter is known
        select=FALSE, Adds extra penalty to reduce beta to zero
        knots=NULL, #Allows you to specify the knots
        sp=NULL, #Vector to supply smoothing parameter
        min.sp=NULL, #Lower boundary of smoothing parameter
        H=NULL, #Quadratic penalty
        gamma=1, #Increaseto >1 to produce smoother models
        fit=TRUE, #Allows gam() to set up model
        paraPen=NULL, #Optional list to specify penalties
        G=NULL,in.out, #For an object call to a previous gam()
        drop.unused.levels=TRUE, #Drop unused levels in fitting
        drop.intercept=NULL, #To exclude an intercept term
        discrete=FALSE, #Used for discrete methods in bam()
        ... #Passing further arguments
        )
```


A Cornucopia of Smooths

The type of smoother is controlled by the bs (basis) argument

The default is a low-rank thin plate spline bs = 'tp'

Many others available

- Cubic splines bs = 'cr' *(Best for big data)*
- P splines bs = 'ps'
- Cyclic splines bs = 'cc' or bs = 'cp' *(Cyclical data)*
- Adaptive splines bs = 'ad'
- Random effect bs = 're'
- Factor smooths bs = 'fs'

- Duchon splines bs = 'ds'
- Spline on the sphere bs = 'sos'
- MRFs bs = 'mrf' *(Markov Random Field)*
- Soap-film smooth bs = 'so'
- Gaussian process bs = 'gp'

The parameter goes in each smooth term

Conditional Distributions

A GAM is just a fancy GLM

Simon Wood & colleagues (2016) have extended the *mgcv* methods to some non-exponential family distributions

- binomial()
- poisson()
- Gamma()
- inverse.gaussian()
- nb() *(Negative Binomial)*
- tw() *(Tweedie)*
- mvn() *(Multivariate Normal)*
- multinom() *(Multinomial)*
- betar() *(Beta)*
- betar() (Beta)
- scat() (Scaled T)
- gaulss() *(Gaussian Location Scale)*
- ziplss() *(Zero Inflation Poisson)*
- twlss() *(Tweedie Location Scale)*
- cox.ph() *(Cox Model for Survival Analysis)*
- gamals() *(Gamma Location Scale)*
- ocat() *(Ordered Categorical)*

Note:
- *Location Scale models allow you to fit the mean & variance*
- *Zero Inflation models allow you to fit zero values observations*

Smooth Interactions

Two ways to fit smooth interactions:

1. Bivariate (or higher order) thin plate splines
   - s(x, z, bs = 'tp')
   - Isotropic; single smoothness parameter for the smooth
   - Sensitive to scales of x and z
   - Same scale in both x and z
2. Tensor product smooths
   - Separate marginal basis for each smooth, separate smoothness parameters
   - Invariant to scales of x and z
   - Use for interactions when variables are in different units
   - te(x, z)
3. Pure Interactions
   - ti() fits pure smooth interactions; where the main effects of x and z have been removed from the basis
   - s(x) + s(z) + ti(x, z)

Factor Smooth Interactions

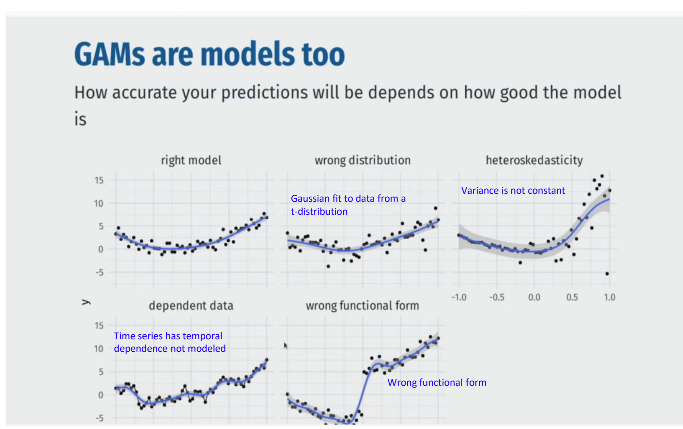
Two ways for factor smooth interactions:

1. by variable smooths
   - entirely separate smooth function for each level of the factor
   - each has its own smoothness parameter
   - centred (no group means) so include factor as a fixed effect
   - y ~ f + s(x, by = f)

2. bs = 'fs' basis
   - smooth function for each level of the function
   - share a common smoothness parameter
   - fully penalized; include group means
   - closer to random effects
   - y ~ s(x, f, bs = 'fs')

Model Checking

How do you know you have the right degrees of freedom?
gam.check()



Model Fit Checklist

Many choices:

- Choice for k
- Choice for distribution family
- Choice for type of smoother
- Missing effects

Page 29

29

Setting Basis the Size of k

Usual 1st Step

- Set k equal to the number of covariates
- e.g. s(x, k=10) or s(x, y, k=100)
- People often choose the defaults
- But should be set to account for wiggliness
- Penalty removes "extra" wigglyness
  - *up to a point!*
- (But computation is slower with bigger k)

Page 30

30

Setting Basis the Size of k

#Checking basis size
norm_model_1 <- gam(y_norm ~ s(x1, k=4) + s(x2, k =4), method = 'REML)
gam.check(norm_model_1)

Compares a random set of residuals to model residuals and if the associations are larger in yours then there is still unmodeled variation the model is not accounting for.

```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.0003467788,0.0005154578]
(score 736.9402 & scale 2.252304).
Hessian positive definite, eigenvalue range [0.000346021,198.5041].
Model rank =  7 / 7

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

        k'  edf k-index p-value
s(x1) 3.00 1.00    0.13  <2e-16 ***
s(x2) 3.00 2.91    1.04    0.83
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

k-index should be close to 1.0. This is the randomization test. This says k for s1 is too small. So, we need to increase the k value for s1.

Note: **edf** equals the number of parameters needed to produce the curve.

Page 31

Setting Basis the Size of k

#Checking basis size
norm_model_2 <- gam(y_norm ~ s(x1, k=12) + s(x2, k =4), method = 'REML)
gam.check(norm_model_2)

```
Method: REML   Optimizer: outer newton
full convergence after 11 iterations.
Gradient range [-5.658609e-06,5.392657e-06]
(score 345.3111 & scale 0.2706205).
Hessian positive definite, eigenvalue range [0.967727,198.6299].
Model rank =  15 / 15

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

         k'   edf k-index p-value
s(x1) 11.00 10.84    0.99    0.38
s(x2)  3.00  2.98    0.86    0.01 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This says k for s2 is too small. So, we need to increase the k value for s2.

Page 32

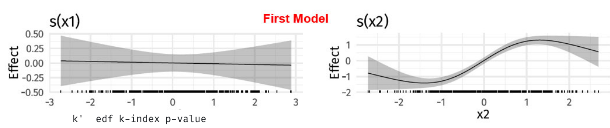
Setting Basis the Size of k

```
#Checking basis size
norm_model_2 <- gam(y_norm ~ s(x1, k=12) + s(x2, k =12), method = 'REML)
gam.check(norm_model_2)
```

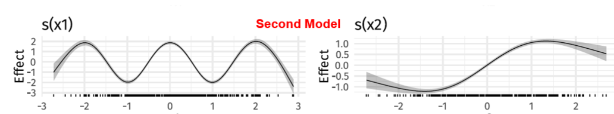
```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-1.136192e-08,6.812328e-13]
(score 334.2084 & scale 0.2485446).
Hessian positive definite, eigenvalue range [2.812271,198.6868].
Model rank =  23 / 23

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

           k'   edf  k-index  p-value
s(x1) 11.00 10.85     0.98     0.31
s(x2) 11.00  7.95     0.95     0.15
```

Things looks pretty good now.

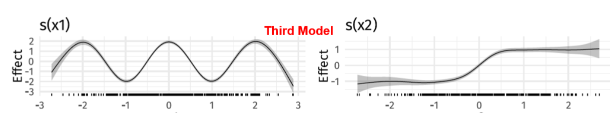
Checking Basis Size




```
          k'  edf k-index p-value
s(x1) 3.00 1.00    0.13  <2e-16 ***
s(x2) 3.00 2.91    1.04    0.83
```




```
           k'   edf k-index p-value
s(x1) 11.00 10.84    0.99    0.38
s(x2)  3.00  2.98    0.86    0.01 **
```




```
           k'   edf k-index p-value
s(x1) 11.00 10.85    0.98    0.31
s(x2) 11.00  7.95    0.95    0.15
```

Continual improvement with the model improvements we just made to k values.

Model Diagnostic Plots

gam.check() plots

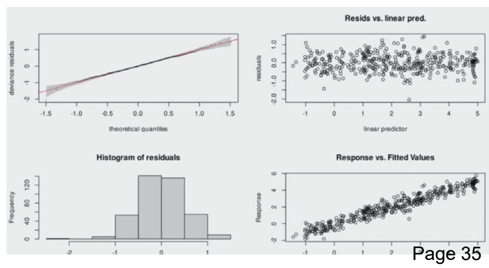
gam.check() creates 4 plots:

1. Quantile-quantile plots of residuals. If the model is right, should follow 1-1 line
2. Histogram of residuals
3. Residuals vs. linear predictor
4. Observed vs. fitted values

gam.check() uses deviance residuals by default

gam.check() for the 3rd model.

Everything looks normal.



Poisson Example

To understand p-values

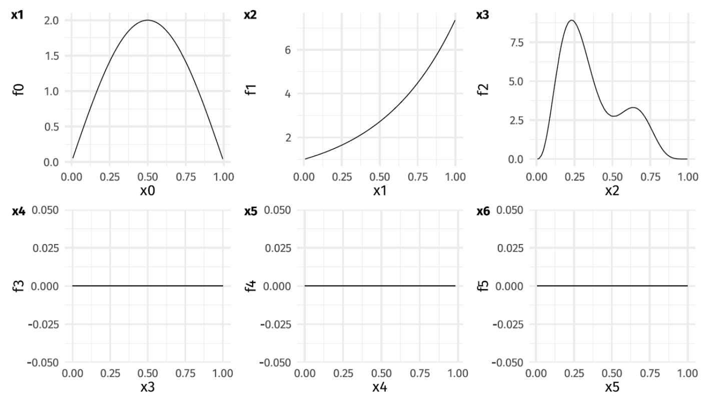
- Simulate Poisson counts
- 4 known functions (left)
- 2 spurious covariates (runif() & not shown)

```
set.seed(3)
n <- 200

#simulate data
dat<-  gamSim(1, n=n, scale=0.15, dist='poisson', verbose = FALSE)
dat <- transform(dat, x4 = runif(n, 0, 1), x5 = runif(n, 0, 1), f4 = rep(0,
n), f5 = rep(0, n)) #x4, f4, x5, f5 are spurious

b <-   gam(y ~ s(x0) + s(x1) + s(x2) + s(x3) + s(x4) + s(x5),data = dat,
family = poisson, method = 'REML', select = TRUE)
```

Turns on the double penalty

For gamSim datasets see: http://web.mit.edu/~r/current/arch/i386_linux26/lib/R/library/mgcv/html/gamSim.html

Poisson Example



p values for smooths

summary(b)

```
Family: poisson
Link function: log

Formula:
y ~ s(x0) + s(x1) + s(x2) + s(x3) + s(x4) + s(x5)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.21758    0.04082   29.83   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
            edf Ref.df  Chi.sq p-value
s(x0) 1.7655088      9   5.264  0.0397 *
s(x1) 1.9271040      9  65.356  <2e-16 ***
s(x2) 6.1351414      9 156.204  <2e-16 ***
s(x3) 0.0002849      9   0.000  0.4068
s(x4) 0.0003044      9   0.000  1.0000
s(x5) 0.1756926      9   0.195  0.2963
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.545   Deviance explained = 51.6%
-REML = 430.78  Scale est. = 1         n = 200
```

k = 10 is the default degrees of freedom

P-value for null hypothesis the true function is a flat function.

p values for smooths

p values for smooths are approximate:

1. They don't account for the estimation of λ_j — treated as known, ***hence p values are biased low – they are lower than they should be.***

2. Rely on asymptotic behavior — they tend towards being right as sample size tends to ∞

3. *The above is also true for Lasso, Ridge, and Elastic Net p-values.*

4. Have the best behavior when smoothness selection is done using ML, then REML.

5. Neither of these are the default, so remember to use method = "ML" or method = "REML" as appropriate

AIC for GAMs

- Comparison of GAMs by a form of AIC is an alternative frequentist approach to model selection

- Rather than using the marginal likelihood, the likelihood of the β_j *conditional* upon λ_j is used, with the EDF replacing k_j , the number of model parameters

- This *conditional* AIC tends to select complex models, especially those with random effects, as the EDF ignores λ_j that are estimated

- Wood et al (2016) suggests a correction that accounts for uncertainty in λ_j

$$AIC = -2\mathcal{L}(\hat{\beta}) + \underbrace{2\text{tr}(\widehat{\mathcal{I}}V'_{\beta})}_{\textit{Trace}}$$

Concurvity in GAMs

- A generalization of co-linearity in GLMs
- The existence of nonlinear dependencies among predictor variables or the existence of non-unique solutions of the system of homogeneous equations.
- Occurs when a smooth term in a model can be approximated by one or more of the other smooth terms in the model.
- Presence of concurvity in the data may lead to poor parameter estimation (upwardly biased estimates of the parameters and underestimation of their standard errors), increasing the risk of committing type I error.
- Detected with a correlation integral: *($z_i = (x_i, y_i)$)*

$$I(r) = \frac{1}{N^2} \sum_{I,j=1}^{N} I(|z_i - z_j| < r).$$

Reference: Amodio, S., Aria, M., & D'Ambrosio, A. (2014). On Concurvity In Nonlinear And Nonparametric Regression Models. *Statistica, 74*, 81-94.

Concurvity Example

```
library(mgcv)

## Simulate data with concurvity...
set.seed(8);n<- 200
f2 <- function(x) {0.2 * x^11 * (10 * (1 - x))^6 + 10 * (10 * x)^3 * (1 - x)^10}
t <- sort(runif(n)) ## first covariate

## Make covariate x a smooth function of t + noise
x <- f2(t) + rnorm(n)*3

cor(x, t) #correlation = -0.4331803

## Simulate response dependent on t and x...
y <- sin(4*pi*t) + exp(x/20) + rnorm(n)*.3

## Fit model...
b <- gam(y ~ s(t, k=15) + s(x, k=15), method="REML")

## Assess concurvity between each term and `rest of model'
concurvity(b)

## Now look at pairwise concurvity between terms...
concurvity(b, full=FALSE)
```

Interpreting the Results

Looking for Values > 0.80. These results pretty look. The correlation between t & x is -0.4331803.

full = TRUE

```
               para       s(t)      s(x)
worst    1.064436e-24 0.60269087 0.6026909
observed 1.064436e-24 0.09576829 0.5728602
estimate 1.064436e-24 0.24513981 0.4659564
```

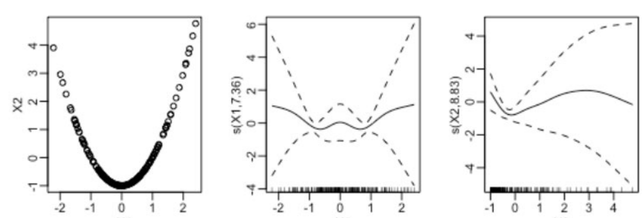
Determines how much each smooth is pre-determined by the others.

full = FALSE

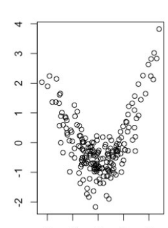
```
$worst
             para         s(t)         s(x)
para 1.000000e+00 7.313872e-26 8.950649e-25
s(t) 7.408676e-26 1.000000e+00 6.026909e-01
s(x) 8.983056e-25 6.026909e-01 1.000000e+00

$observed
             para         s(t)         s(x)
para 1.000000e+00 4.557228e-28 1.704959e-32
s(t) 7.408676e-26 1.000000e+00 5.728602e-01
s(x) 8.983056e-25 9.576829e-02 1.000000e+00

$estimate
             para         s(t)         s(x)
para 1.000000e+00 6.993809e-29 3.458685e-27
s(t) 7.408676e-26 1.000000e+00 4.659564e-01
s(x) 8.983056e-25 2.451398e-01 1.000000e+00
```

Use this mode if the first reveals a high worst-case value to identify where the problem is. Shows the degree to which each variable is pre-determined by each other variable rather than all the other variables.

A Problematic Example



Shows the concurvity effects on the confidence intervals.



```
concurvity(m1, full = TRUE)
```

	para	s(X1)	s(X2)
worst	0	0.84	0.84
observed	0	0.22	0.57
estimate	0	0.28	0.60

Best Practice: Examine visual relationship between two variables.

References

- Marra & Wood (2011) Computational Statistics and Data Analysis 55 2372–2387.
- Marra & Wood (2012) Scandinavian Journal of Statistics, Theory and Applications 39(1), 53–74.
- Nychka (1988) Journal of the American Statistical Association 83(404) 1134–1143.
- Wood (2017) Generalized Additive Models: An Introduction with R. Chapman and Hall/CRC. (2nd Edition)
- Wood (2013a) Biometrika 100(1) 221–228.
- Wood (2013b) Biometrika 100(4) 1005–1010.
- Wood et al (2016) JASA 111 1548–1563