Generalized Additive Models (GAMs) with R

Examples and Features of GAMs

Day 4 Agenda (1 of 2)

- Theory versus 'Real World'?
- Pick up Theory 'Along the Way'
- Cherry Trees Again . . . *Each purple line is a different example*
  - **Fit another GAM example with smooths of single variables.**
  - **Use Thin Plate Regression Splines (TPRS).**
  - **Vary the Dimension, *k*.**
  - **Vary value of γ for GCV or UBRE scores.**
  - Advantages, disadvantages of smoothing bases.
  - **Smooths of several variables: TPRS.**
  - **Smooths of several variables: Tensor products.**
  - Comparison of Tensor Product and TPRS.
  - **Mixing parametric (factor) and smooth model terms.**

Day 4 Agenda (2 of 2)

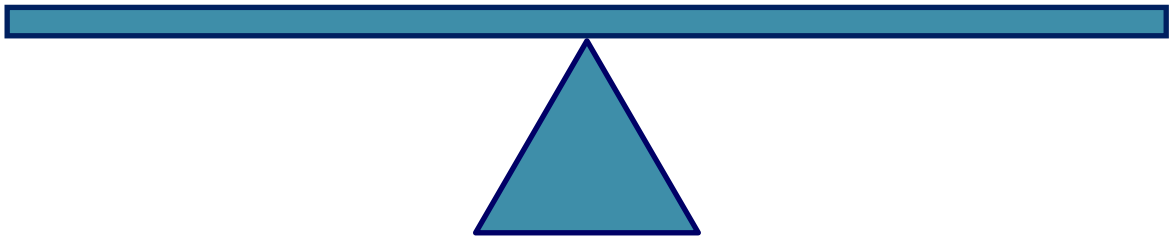
- Brain Imaging Example

  Each purple line is a different example

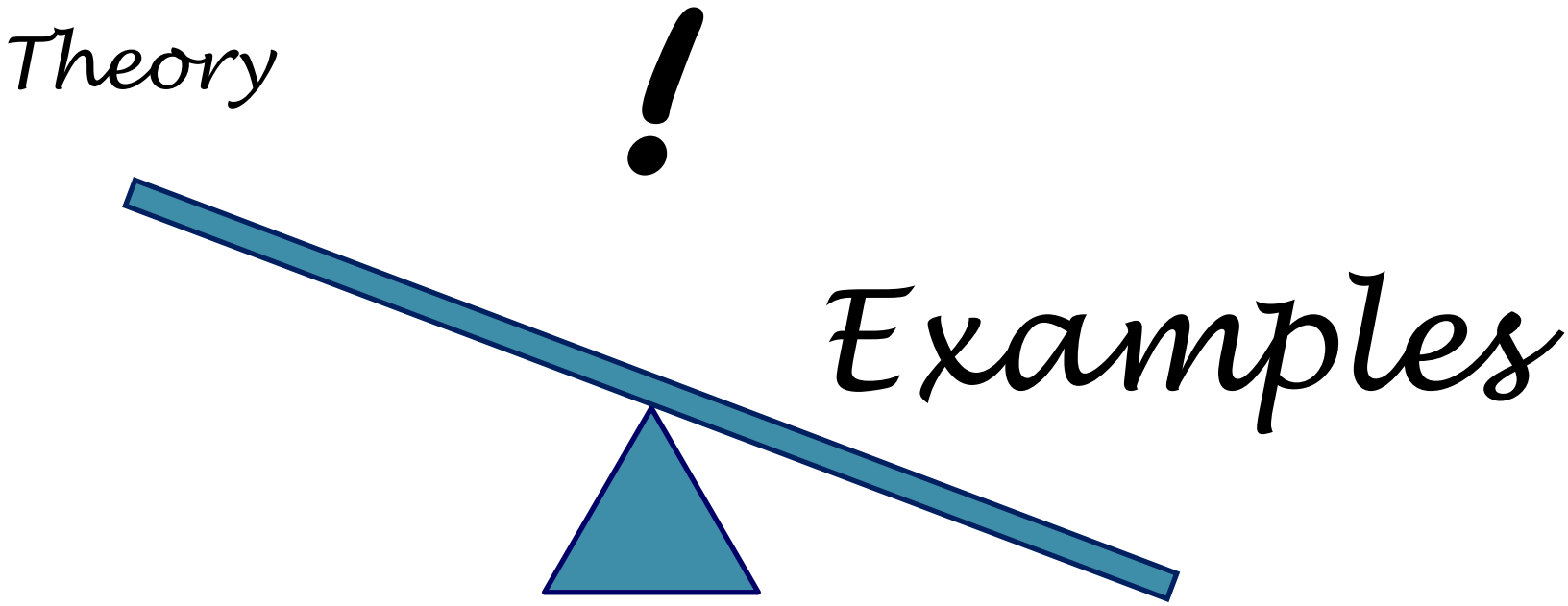
  - **Model medFPQ ~ smooths of Y, X without transformation**
  - **Model m1 ~ isotropic smooth of Y, X with power transform**
  - **Model m2 ~ isotropic smooth of Y, X with Gamma structure**
  - **Model m3 ~ additive structure**
  - **Model m4 ~ replaces m2 with completely nested structure**
  - **Model tm ~ tensor product smooth in additive**
  - **Model tm1~ isotropic smooth in additive**
- Air Pollution in Chicago Example
  - **Model ap0~ poisson, only one predictor (time) is smoothed**
  - **Model ap1~ smooth all predictor variables**
  - **Model ap2~ transform data, use 3-way tensor product**
  - **Model ap3~ transform data, use 2-way te() term**
- Other GAM packages in R

Theory versus ‘The Real World’ ?

Theory ? *Examples*



Theory versus ‘The Real World’ ?



GAM Theory Things We Missed . . .

- Other (Cubic Spline) Smoothing Bases:
  - Cyclic cubic regression spline
  - P-splines
  - Thin plate regression splines
  - Shrinkage smoothers
- Setting up GAMs as Penalized GLMs
- Justifying P-IRLS
- Degrees of Freedom and Residual Variance Estimation
- Smoothing Parameter Selection Criteria
  - Known scale parameter: UBRE
  - Unknown scale parameter: Cross Validation
    - Problems with ordinary cross validation
  - Generalized cross validation
  - GVC/UBRE/AIC in the generalized case

But ! ! ! We will revisit many of these in the examples . . .

Cherry Trees Again . . .

- We redo the example from Day 3:

```
> library(mgcv)
> data(trees)
> ct1 <- gam(Volume~s(Height)+s(Girth),
          family=Gamma(link=log),data=trees)
```

- Fitting the Generalized Additive Model:

$$\log(\mathrm{E}[\mathrm{Volume}_i]) = f_1(\mathrm{Height}_i) + f_2(\mathrm{Girth}_i)$$

where Volume ~ Gamma and f_j are smooth functions.

  - Degree of smoothness of f_j is estimated by GCV.

Cherry Trees Again . . .

- We print and plot:

```
> ct1

Family: Gamma
Link function: log

Formula:
Volume ~ s(Height) + s(Girth)

Estimated degrees of freedom:
1.0000  2.4222  total = 4.422254

GCV score: 0.008082356

> par(mfrow=c(1,2))
> plot(ct1,residuals=TRUE)
```

1 DoF for Height

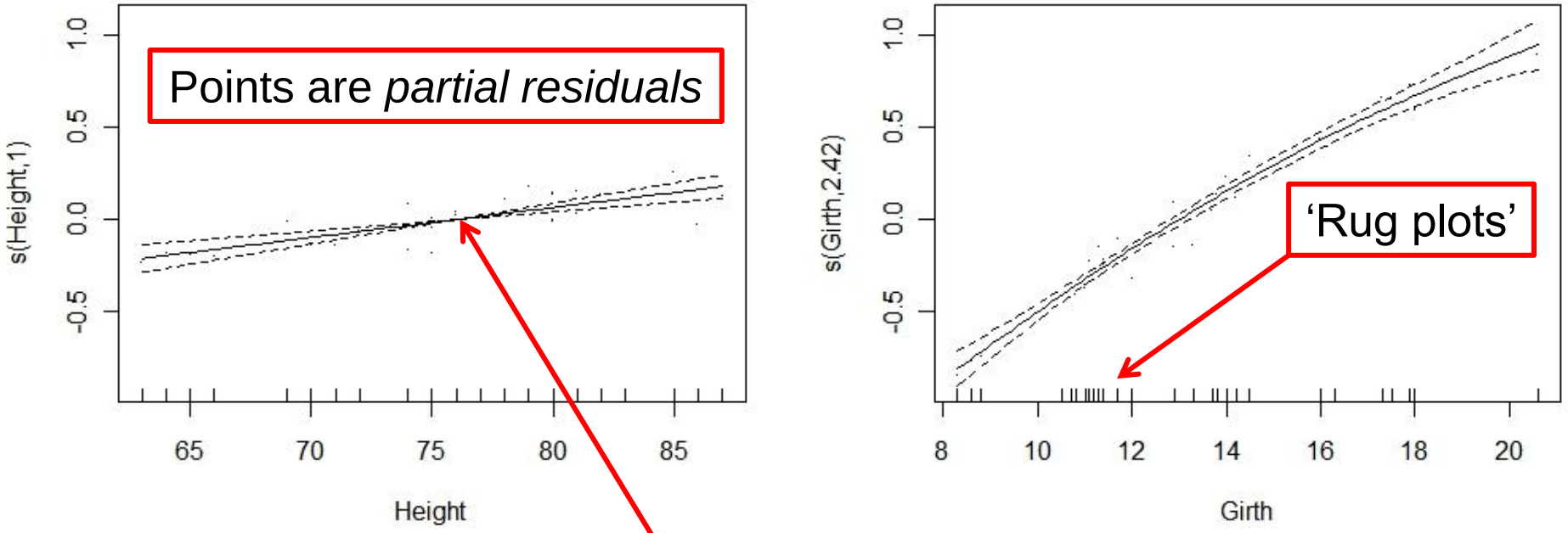
2.42 DoF for Girth

1+2.42+1 = 4.42 total DoF

Cherry Trees Again . . .

- Components of GAM model fits to cherry tree data:

Dashed lines are confidence intervals



No uncertainty where line passes through zero

Finer Control of GAM . . . Basis

- **`gam()`** call producing ct1 had many options set to default:
  - Default basis is thin plate regression splines.

```
> ct2 <- gam(Volume~s(Height,bs="cr")+s(Girth,
   bs="cr"),family=Gamma(link=log),data=trees)

> ct2

Family: Gamma
Link function: log

Formula:
Volume ~ s(Height, bs = "cr") + s(Girth, bs = "cr")

Estimated degrees of freedom:
1.0000 2.4188  total = 4.418778

GCV score: 0.008080514
```

Penalized cubic regression spline basis

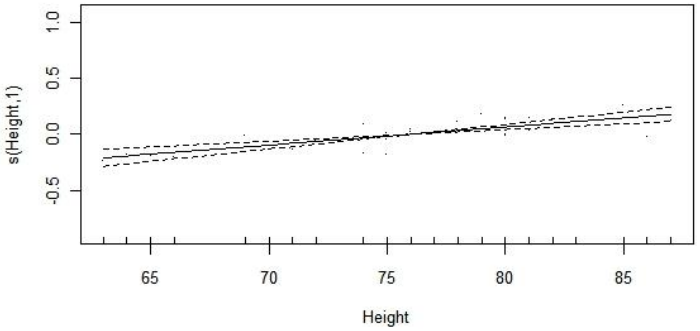
Finer Control of GAM . . . Basis

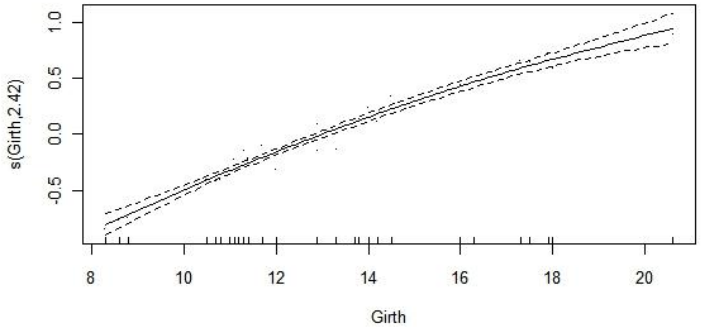
- plot ct1 and ct2 next to each other:

```
> par(mfrow=c(2,2))
> plot(ct1,residuals=TRUE)
> plot(ct2,residuals=TRUE)
```

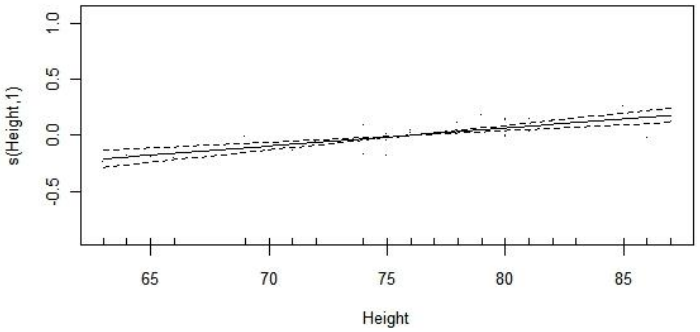
Plots are indistinguishable

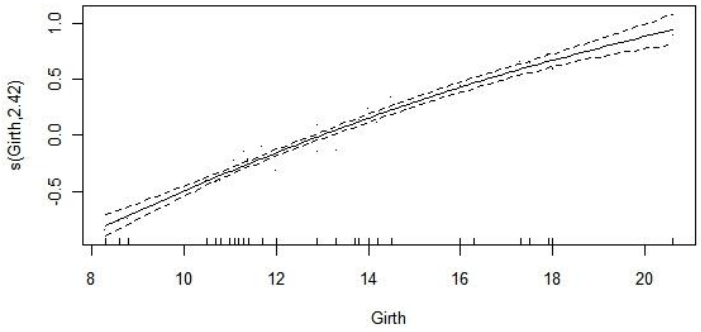
ct1





ct2





Finer Control of GAM . . . Dimension, k

```
> ct3 <- gam(Volume ~ s(Height)
          + s(Girth,bs="cr",k=20),
          family=Gamma(link=log),data=trees)
> ct3

Family: Gamma
Link function: log

Formula:
Volume ~ s(Height) + s(Girth, bs = "cr", k = 20)

Estimated degrees of freedom:
1.0000 2.4243  total = 4.424292

GCV score: 0.008082974
```

Finer Control of GAM . . . Gamma, γ

- Final default choice, value of γ in GCV or UBRE scores optimized to select degree of smoothness of each term
  - Default value is 1 which can cause overfitting.
  - Using $\gamma \approx 1.4$ has been suggested as corrective without compromising model fit.

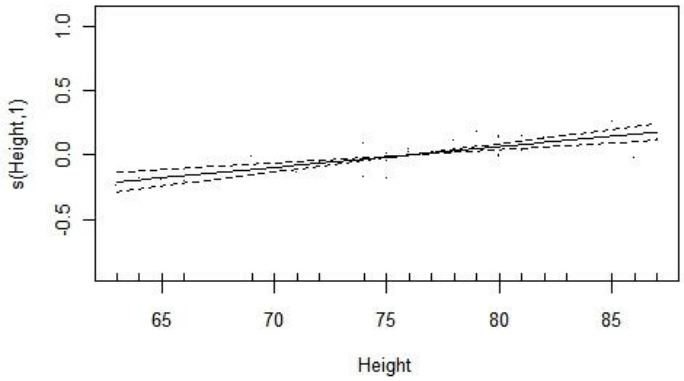
```
> ct4 <- gam(Volume ~ s(Height) + s(Girth),
             family=Gamma(link=log),data=trees,gamma=1.4)
> ct4

Family: Gamma
Link function: log
Formula:
Volume ~ s(Height) + s(Girth)
Estimated degrees of freedom:
1.0000 2.1696  total = 4.169632
GCV score: 0.009228008
```

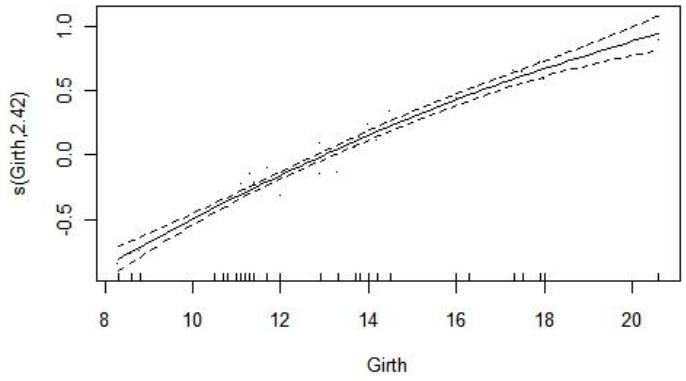
Fewer DoF

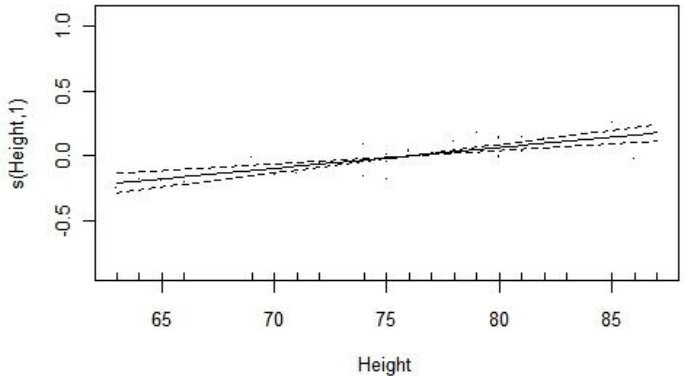
Finer Control of GAM . . . Gamma, $\gamma \approx 1.4$

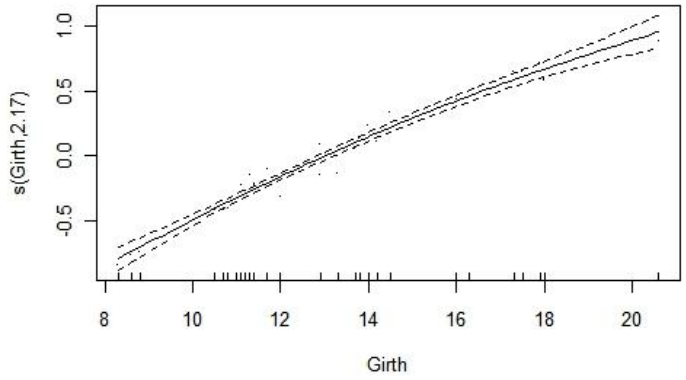
```
> par(mfrow=c(2,2))
> plot(ct1,residuals=TRUE)
> plot(ct4,residuals=TRUE)
```

Plots 2 rows, 2 column in a frame









Smooths of More Than One Predictor

Two terms

- **`gam()`** can model smooths of more than one predictor:
  $\log(\mathrm{E}[\mathrm{Volume}_i]) = f(\mathrm{Height}_i, \mathrm{Girth}_i)$ where Volume_i ~ Gamma
- First function f is a smooth with an ***isotropic thin plate regression spline***:

```
> ct5 <- gam(Volume ~ s(Height,Girth,k=25),
                family=Gamma(link=log),data=trees)
> ct5

Family: Gamma
Link function: log
Formula:
Volume ~ s(Height, Girth, k = 25)
Estimated degrees of freedom:
4.7911  total = 5.791102
GCV score: 0.009357594
```

Smooths of More Than One Predictor

- **`gam()`** can model smooths of more than one predictor:

$$\log(\mathrm{E}[\mathrm{Volume}_i]) = f(\mathrm{Height}_i, \mathrm{Girth}_i)$$ where Volume_i ~ Gamma

- Then we model function f as a ***tensor product smooth***:

```
> ct6 <- gam(Volume ~ te(Height,Girth,k=5),
                 family=Gamma(link=log),data=trees)
> ct6

Family: Gamma
Link function: log
Formula:
Volume ~ te(Height, Girth, k = 5)
Estimated degrees of freedom:
3.000175  total = 4.000175
GCV score: 0.008197151
```

Two terms still

Dimension for each marginal basis

Only 3 DoF amounts to this model

$$\log(\mathrm{E}[\mathrm{Volume}_i]) = \beta_0 + \beta_1 \mathrm{Height}_i + \beta_2 \mathrm{Girth} + \beta_3 \mathrm{Height}_i \mathrm{Girth}_i$$

Advantages, Disadvantages of Package `mgcv` Smoothing Bases

bs	Description	Advantages	Disadvantages
“tp”	Thin plate re-gression splines (TPRS)	Can smooth w.r.t. any number of covariates. Invariant to rotation of covariate axes. Can select penalty order. No ‘knots’ and some optimality properties.	Computationally expensive for large data sets. Not invariant to covariate rescaling.
“ts”	TPRS with shrinkage	As TPRS, but smoothness selection can zero term completely.	As TPRS.
“cr”	Cubic regression spline (CRS)	Computationally cheap. Directly interpretable parameters.	Can only smooth w.r.t. on covariate. Knot based. Doesn’t have TPRS optimality.

Advantages, Disadvantages of Package `mgcv` Smoothing Bases

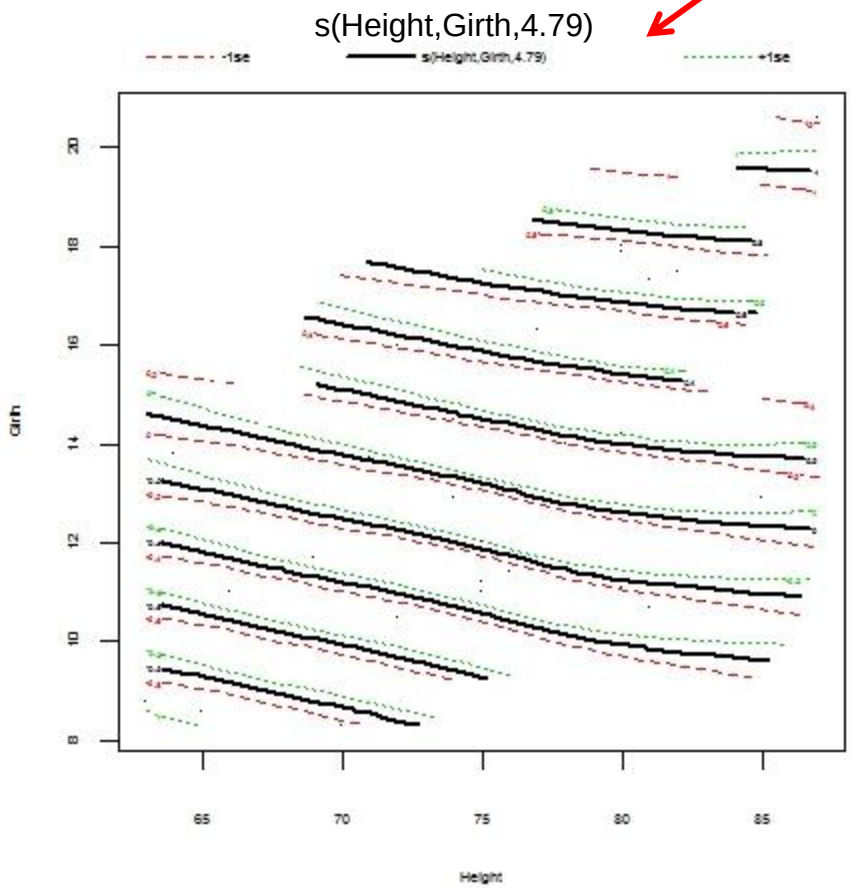
bs	Description	Advantages	Disadvantages
“cs”	CRS with shrinkage	Can smooth w.r.t. any number of covariates. Invariant to rotation of covariate axes. Can select penalty order. No ‘knots’ and some optimality properties.	As CRS.
“cc”	Cyclic CRS	As CRS, but start point same as end point.	As CRS.
“ps”	P-splines	Any combination of basis and penalty order possible. Perform well in tensor products.	Based on equally spaced knots. Penalties awkward to interpret. No optimality properties available.

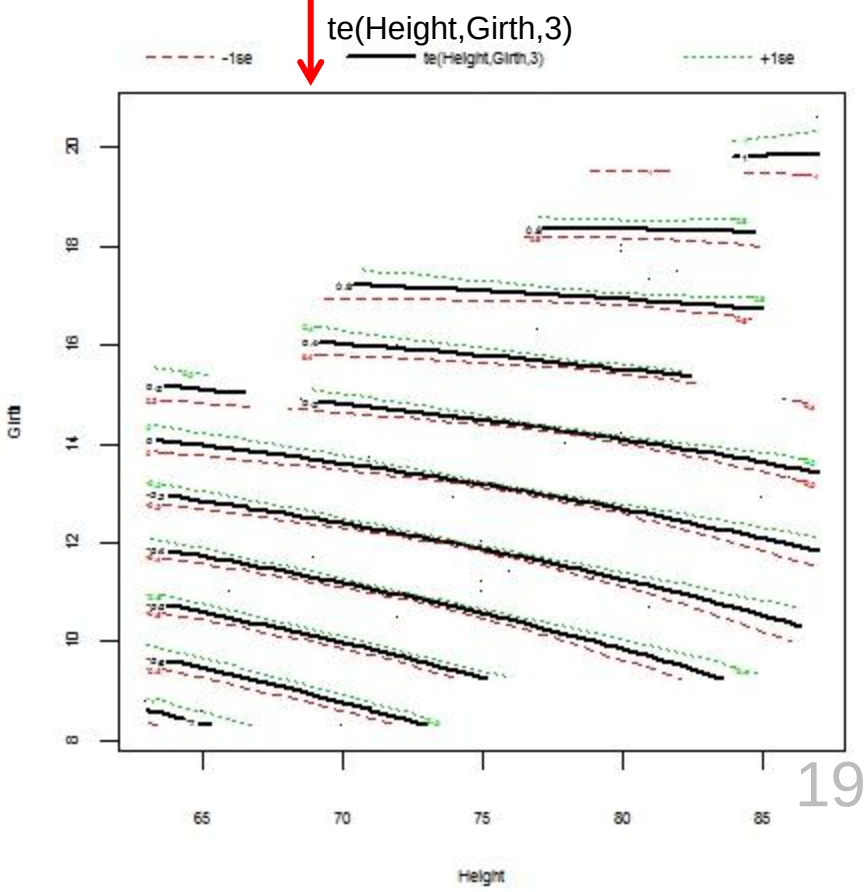
Plots of Thin Plate and Tensor Product Regression Spline Fits

```
> par(mfrow=c(1,2))
> plot(ct5,too.far=0.15)
> plot(ct6,too.far=0.15)
```

Thin Plate Regression Spline Fit

Tensor Product Spline Fit





Tensor Product (`te`) versus Thin Plate Regression Spline (`s`)

Tensor product, `te`	**TPRS, `s(…,bs="tp")`**
Invariant to linear rescaling of covariates, but not to rotation of covariate space.	Invariant to rotation of covariate space (isotropic), but not to rescaling of covariates.
Good for smooth interactions of quantities measured in different units, or where very different degrees of smoothness appropriate relative to different covariates.	Good for smooth interactions of quantities measured in same units, such as spatial co-ordinates, where isotropy is appropriate.
Computationally inexpensive, provided TPRS bases are not used as marginal bases.	Computational cost can be high as it increase with square of number of data (can be avoided by approximation).
Apart from scale invariance, not much supporting theory.	Some optimality results available.

Parametric Model Terms

- For example, given that smooth of Height in **ct1** is estimated as a straight line, we might as well fit the model:

```
> gam(Volume ~ Height+s(Girth),
                family=Gamma(link=log),data=trees)
```

Create a factor variable

- But with **Height** as categorical variable:

```
> trees$Hclass <- factor(floor(trees$Height/10)-5,
                         labels=c("small","medium","large"))
```

- Now we fit and plot a **gam** using **Hclass** as factor variable

```
> ct7 <- gam(Volume ~ Hclass+s(Girth),
             family=Gamma(link=log),data=trees)
> par(mfrow=c(1,2))
> plot(ct7,all.terms=T)
```

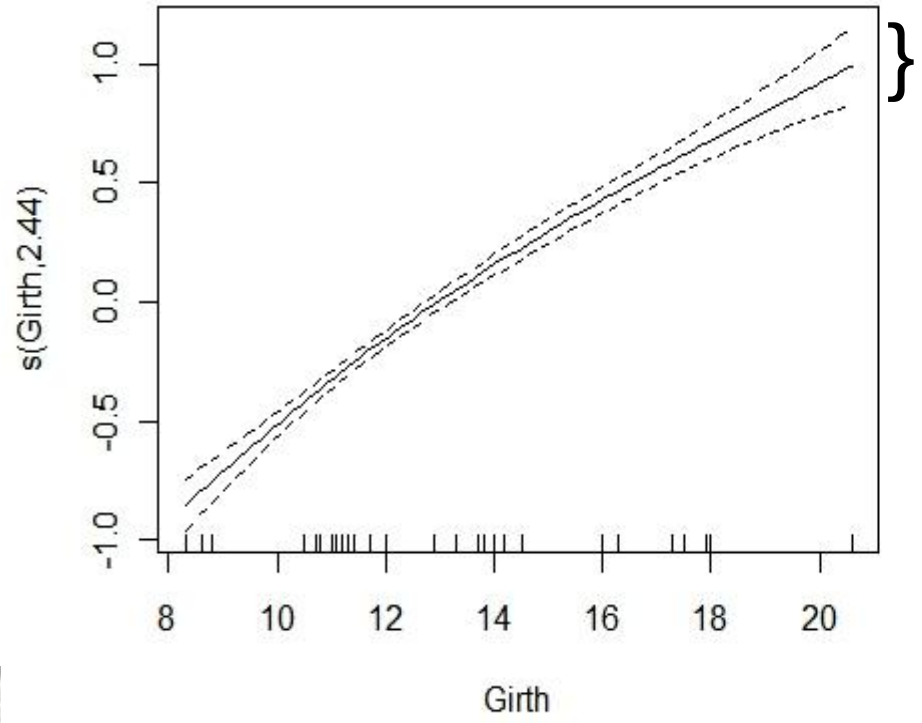
Plot of `ct7`, semi-parametric model

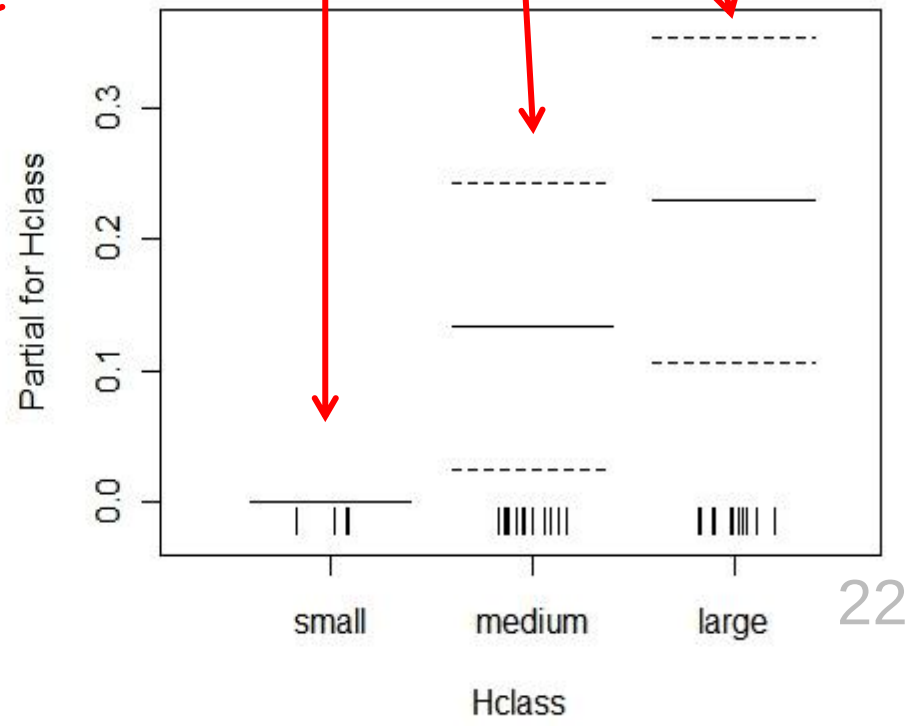
```
> par(mfrow=c(1,2))
> plot(ct7,all.terms=T)
```

95% Confidence Interval

Estimated Effect For Each Level

smooth of Girth





Parametric Model Terms

```
> anova(ct7)
Family: Gamma
Link function: log

Formula:
Volume ~ Hclass + s(Girth)

Parametric Terms:
        df      F p-value
Hclass   2 6.965 0.00385

Approximate significance of smooth terms:
           edf Ref.df     F p-value
s(Girth) 2.444  3.076 156.2  <2e-16
```

Parametric Model Terms

```
> AIC(ct7)
[1] 154.9267
> summary(ct7)
Family: Gamma
Link function: log
Formula:
Volume ~ Hclass + s(Girth)

Parametric coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       3.12722    0.04841  64.593  < 2e-16 ***
Hclass[T.medium]  0.13415    0.05469   2.453 0.021332 *
Hclass[T.large]   0.23000    0.06180   3.722 0.000982 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
           edf Ref.df     F p-value
s(Girth) 2.444  3.076 156.2  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.967   Deviance explained = 96.9%
GCV score = 0.012075  Scale est. = 0.0099548  n = 31
```

Significance of individual levels of parametric term

New terms

Brain Imaging Example

- Data from brain imaging by **functional magnetic resonance scanning** (Landau et al. 2003)
  - Data frame **`brain`** in package **`gamair`**.

```
> library(gamair)
> data(brain)
> brain
      X  Y    medFPQ region meanTheta
1    65  9  3.923048     NA  2.838555
2    60 10  0.492985      0  1.218145
3    63 10  2.759576     NA  1.983947
4    65 10  0.000003      0  1.306030
5    66 10  0.854175      0 -2.496284
6    54 11  2.235795     NA  1.648068
.     .  .         .      .         .
.     .  .         .      .         .
## Each row corresponds to one voxel (Wikipedia)
```

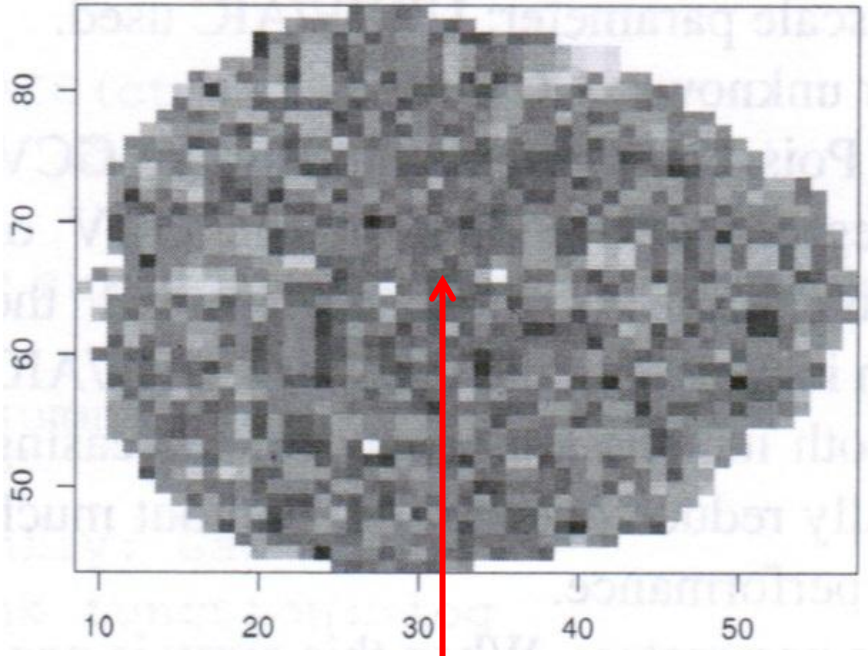
Voxel locations

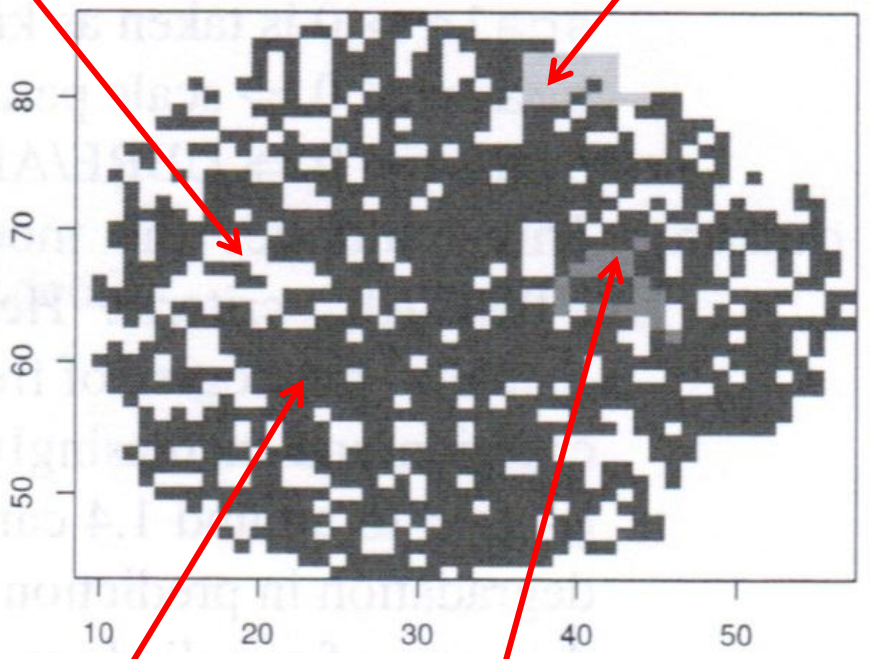
Brain activity level

Voxel region

medFPQ ~ X, Y

Raw Brain Imaging Data





Unclassified voxels

Experimentally stimulated

Median of 3 brain activity measurements

Base region

Region of experimental interest

medFPQ is ‘noisy,’ try to partition into (1) smooth trend; and (2) ‘random variability’ components

Main `gam` Arguments for Controlling Smoothness Estimation Process

`scale`	Controls use of UBRE/AIC or GCV for smoothing parameter selection. `scale > 0:` known scale parameter: use UBRE/AIC. `scale < 0:` unknown scale parameter: use GCV. `scale = 0:` UBRE/AIC for Poisson or binomial, otherwise GCV.
`gamma`	Multiplies model degrees of freedom in GCV or UBRE/AIC criteria. As gamma increased from 1, the 'penalty' per degree of freedom increases in the GCV or UBRE/AIC criterion , producing increasingly smooth models. Increasing gamma to ~1.4 can reduce over-fitting without affecting prediction error performance.
`sp`	Supplied array of smoothing parameters. Negative (non-negative) elements suppress (are used as) smoothing parameters for corresponding term(s). Useful for selectively controlling the smoothness of some terms.
`min.sp`	Can supply minimum values for smoothing parameters with min.sp argument.
`method`	Argument usually set with gam.method, which controls numerical method used to estimate smoothing parameters.

Brain Imaging Outliers

- Will model **`medFPQ`** as function of **`X`** and **`Y`**
  - Get rid of two outliers: values of 3 x 10^{-6} and 4 x 10^{-7}
  - Other 1565 voxels range from 0.003 and 20.

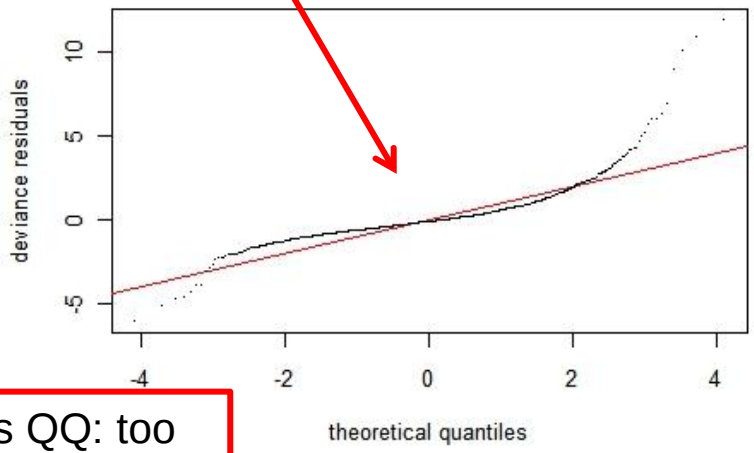
```
> brain <- brain[brain$medFPQ>=3e-5,] #exclude 2 outliers
```

- Then try Gaussian model without transformation:

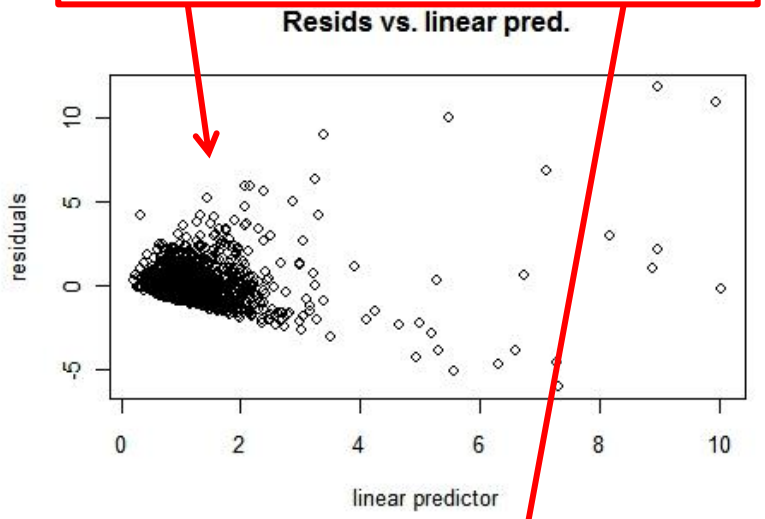
```
> m0 <- gam(medFPQ~s(Y,X,k=100),data=brain)
> gam.check(m0)

Smoothing parameter selection converged after 6 iterations.
The RMS GCV score gradiant at convergence was 5.724028e-05 .
The Hessian was positive definite.
The estimated model rank was 100 (maximum possible: 100)
```

Initial Brain Imaging Plots (model `m0`)

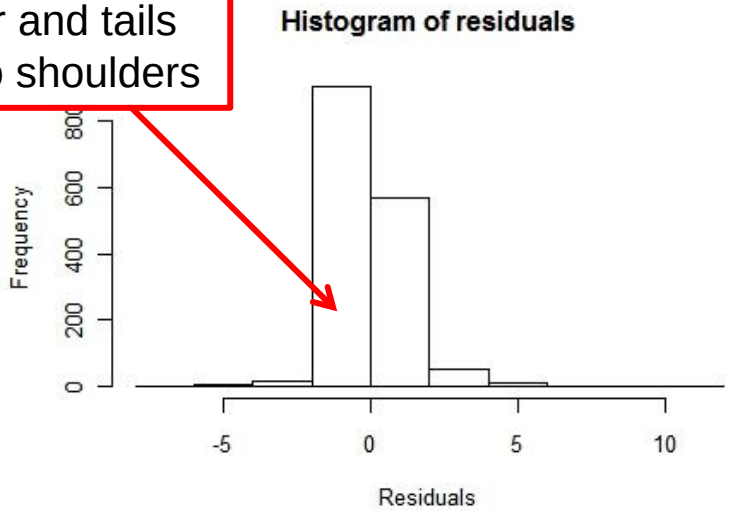
Evidently not Gaussian

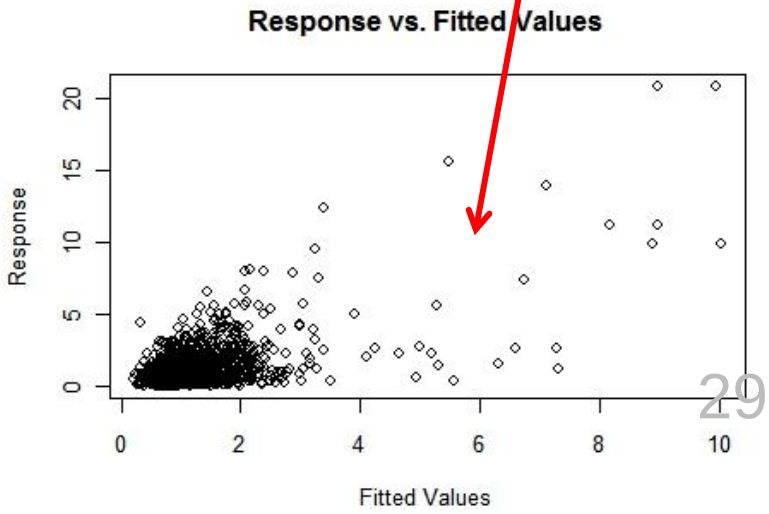


Evidence of non-constant variance



Confirms QQ: too many residuals in center and tails relative to shoulders





How is Variance Increasing with the Mean?

- If we assume $\text{var}(y_i) \propto \mu_i^{\beta}$ where $u_i = \text{E}(y_i)$ and β is some parameter, then we can estimate β with regression:

```
> e <- residuals(m0); fv <- fitted(m0)
> lm(log(e^2)~log(fv))

Call:
lm(formula = log(e^2) ~ log(fv))

Coefficients:
(Intercept)       log(fv)
     -1.959         1.919
```

Could use log transform of medFPQ, but settle on fourth root transform and a log link function

So $\beta \approx 2$

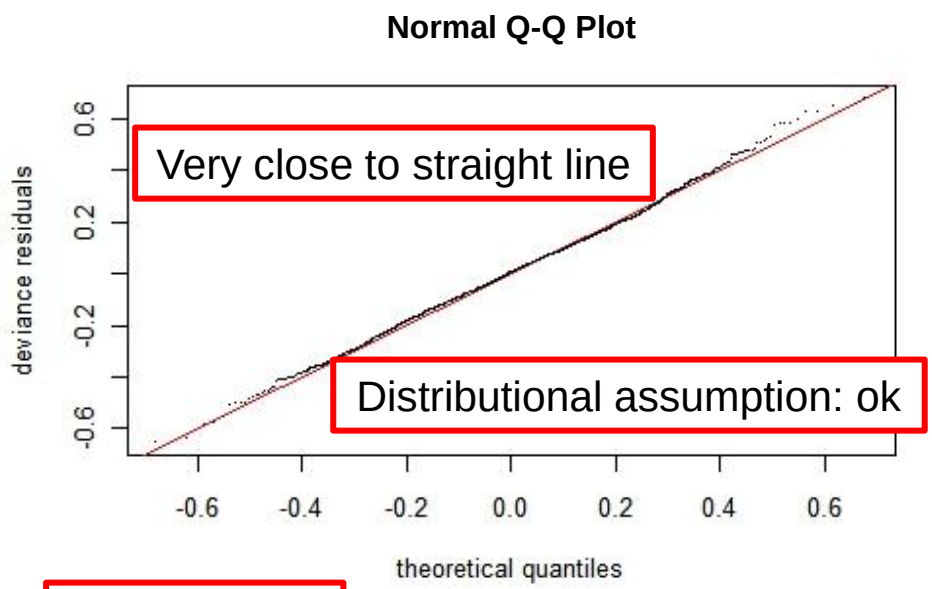
Second Model Fit

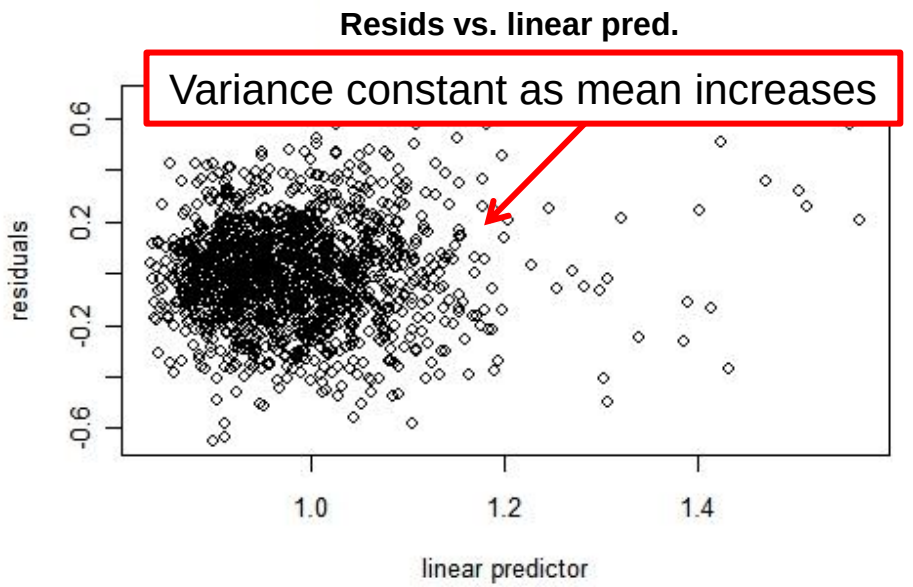
- The following fits the model, based first on transforming the data, and then on the use of the Gamma distribution:

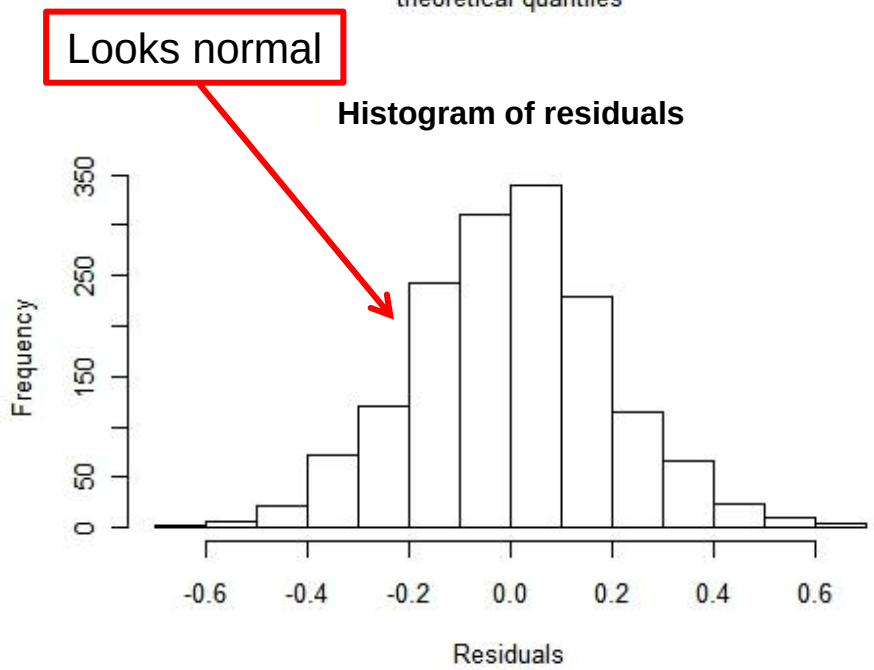
```
> m1<-gam(medFPQ^.25~s(Y,X,k=100),data=brain)
> gam.check(m1)
> m2<-gam(medFPQ~s(Y,X,k=100),
        data=brain,family=Gamma(link=log),
        optimizer="perf")
> mean(fitted(m1)^4)
> mean(fitted(m2));mean(brain$medFPQ)
[1] 0.9838708 # m1 tends to under-estimate
[1] 1.211685  # m2 is substantially better
[1] 1.249506  # this is actually the mean
```

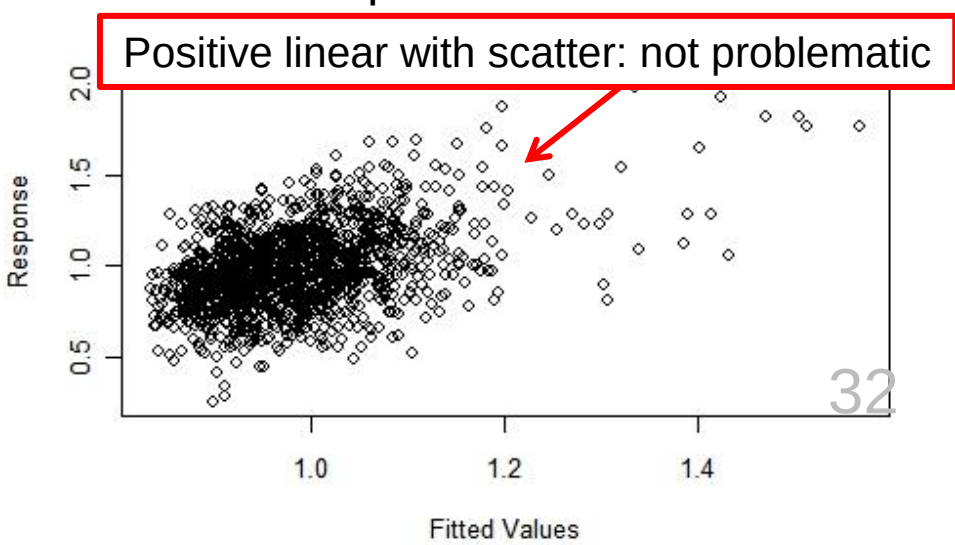
Plots from gam.check on next slide

Model Checking Plots (`m1`)









Closer Look at model m2

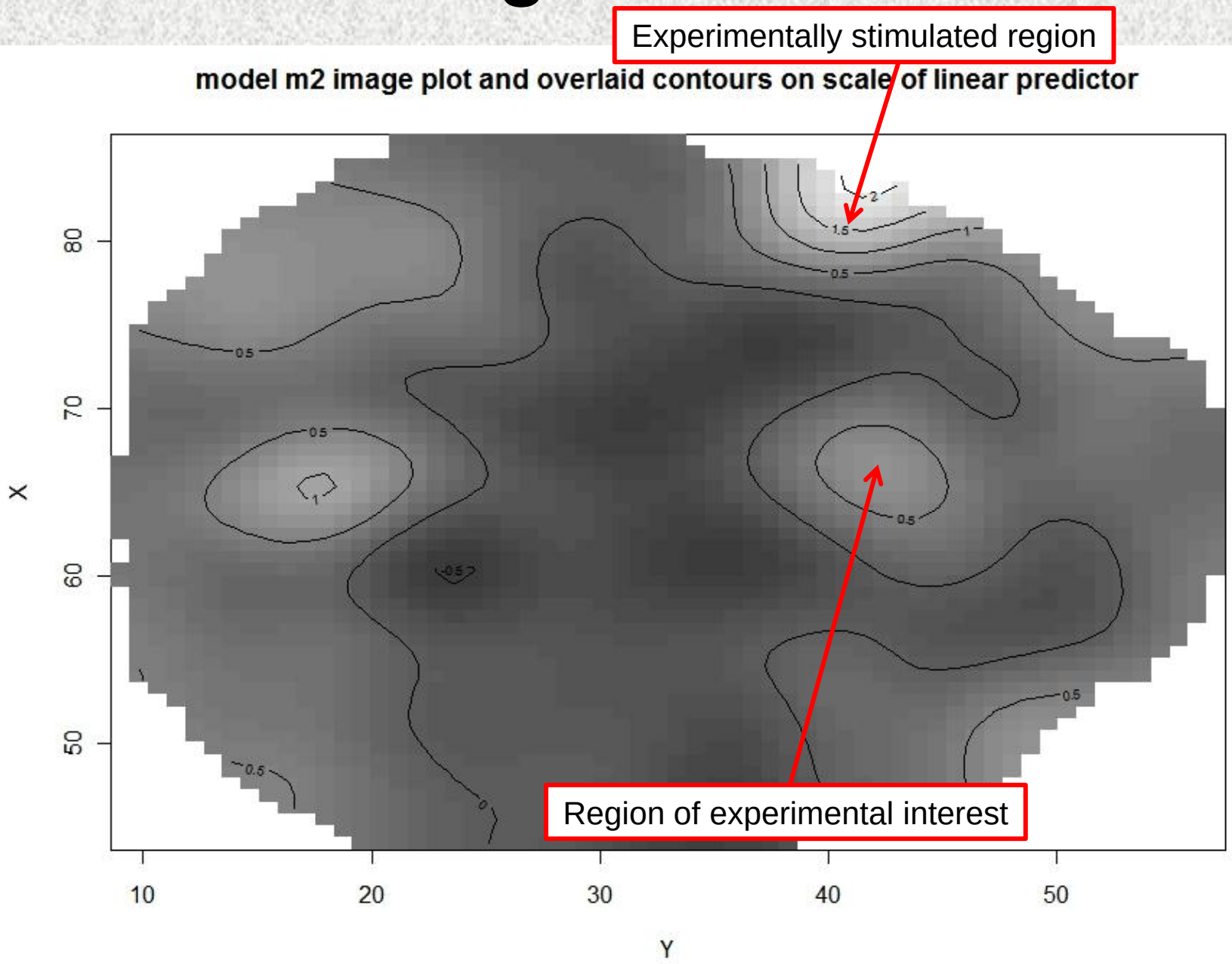
- Best model seems to be Gamma log link **m2**:

```
> m2
Family: Gamma
Link function: log

Formula:
medFPQ ~ s(Y, X, k = 100)

Estimated degrees of freedom:
63.01  total = 64.00951

GCV score: 0.6006784
```

Too many DoF?

```
> vis.gam(m2,plot.type="contour",too.far=0.03,
        color="gray",n.grid=60,zlim=c(-1,2),main="model
        m2 image plot and overlaid contours on scale of
        linear predictor")
```

Model m2 Image Plot with Contours



Is an Additive Structure Better?

- Large number of DoF in model m2 raises question of a better fit with a different, simpler model structure:

$$\log\left(\mathrm{E}\left[\mathrm{medFPQ}_i\right]\right) = f_1\left(\mathrm{Y}_i\right) + f_2\left(\mathrm{X}_i\right), \text{..where...medFPQ} \sim \text{Gamma}$$

```
> m3 <- gam(medFPQ~s(Y,k=30)+s(X,k=30),data=brain,
            family=Gamma(link=log),optimizer="perf")
> m3
Family: Gamma
Link function: log

Formula:
medFPQ ~ s(Y, k = 30) + s(X, k = 30)

Estimated degrees of freedom:
 9.5515 19.6950  total = 30.24652

GCV score: 0.6825806
```

m2 deviance explained = 25.5%
m3 deviance explained = 19.4%

GCV higher

Compare with Analysis of Deviance

```
> anova(m3,m2,test="F")
Analysis of Deviance Table

Model 1: medFPQ ~ s(Y, k = 30) + s(X, k = 30)
Model 2: medFPQ ~ s(Y, X, k = 100)
  Resid. Df Resid. Dev     Df Deviance     F    Pr(>F)
1    1534.8     979.79
2    1501.0     903.32 33.763   76.462 3.931 5.628e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

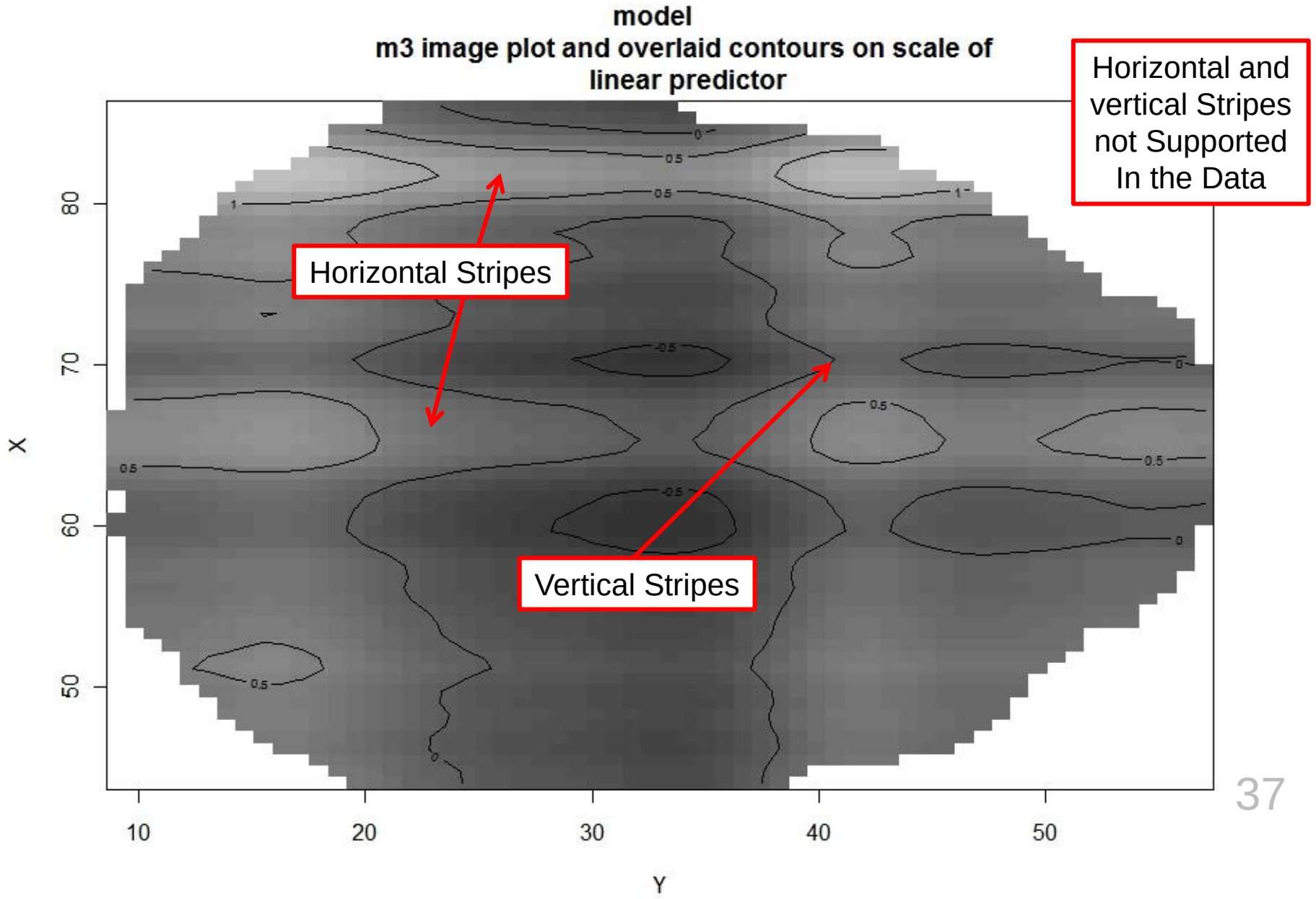
Additive Model Rejected

Looks Better

```
m4 <- gam(medFPQ~s(Y,k=30)+s(X,k=30)+s(Y,X,k=100),data=brain,
      family=Gamma(link=log),optimizer="perf")

# Compare models m3 and m4
anova(m3,m4,test="F")
> vis.gam(m3,plot.type="contour",too.far=0.03,
        color="gray",n.grid=60,zlim=c(-1,2),main="model
        m2 image plot and overlaid contours on scale of
        linear predictor")
```

Same Result, m4 Better

Additive Model m3 Image Plot



Isotropic or Tensor Product Smooths?

- Is worth checking what results look like if we use a scale invariant tensor product smooth of Y and X.
  - Are computationally efficient
- For example:

```
tm <- gam(medFPQ~te(Y,X,k=10),data=brain,family=
      Gamma(link=log), optimizer="perf")
tm1 <-gam(medFPQ~s(Y,k=10,bs="cr")+s(X,bs="cr",k=10),
      data=brain,family=Gamma(link=log),optimizer="perf")
```

Tensor product of two cubic regression spline bases, rank 10 x rank 10 = rank 100

Purely additive model with two rank 10 cubic regression splines so tm1 nested within tm

- Fits a tensor product smooth to FPQ data, storing result in **`tm`** and then fits a purely additive model, storing results in **`tm1`**.

Isotropic or Tensor Product Smooths?

- As with previous models, comparison of GCV scores suggests additive model not the best:

```
> tm1

Family: Gamma
Link function: log

Formula:
medFPQ ~ s(Y, k = 10, bs = "cr") + s(X, bs = "cr", k = 10)

Estimated degrees of freedom:
8.4569 8.0810  total = 17.53793

GCV score: 0.7399581
```

Additive Model

```
> tm

Family: Gamma
Link function: log

Formula:
medFPQ ~ te(Y, X, k = 10)

Estimated degrees of freedom:
58.009  total = 59.0086

GCV score: 0.6175363
```

Tensor Product

Looks Better

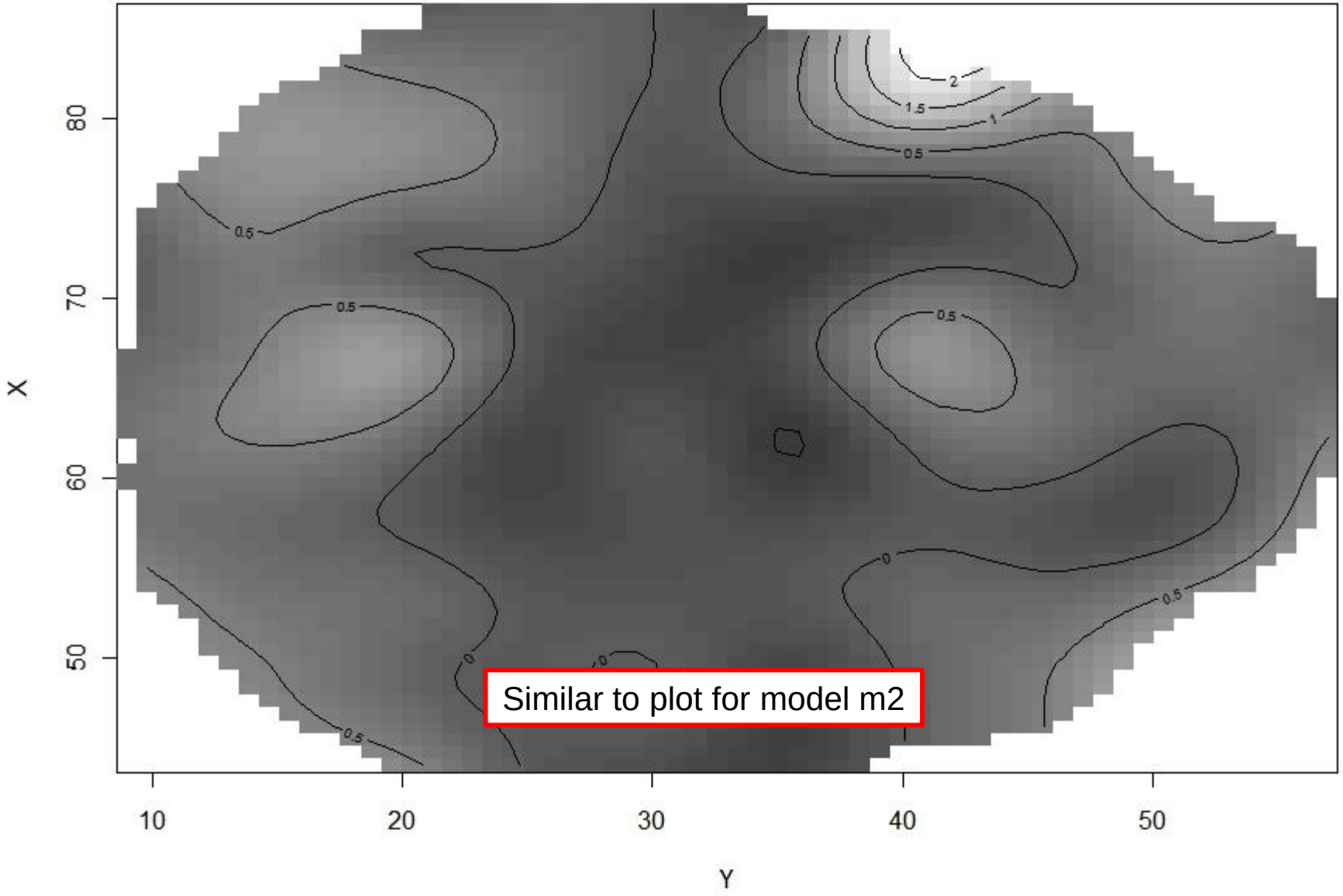
Isotropic or Tensor Product Smooths?

- Similarly, an approximate test of the null hypothesis that the additive structure is appropriate suggests accepting **tm:**

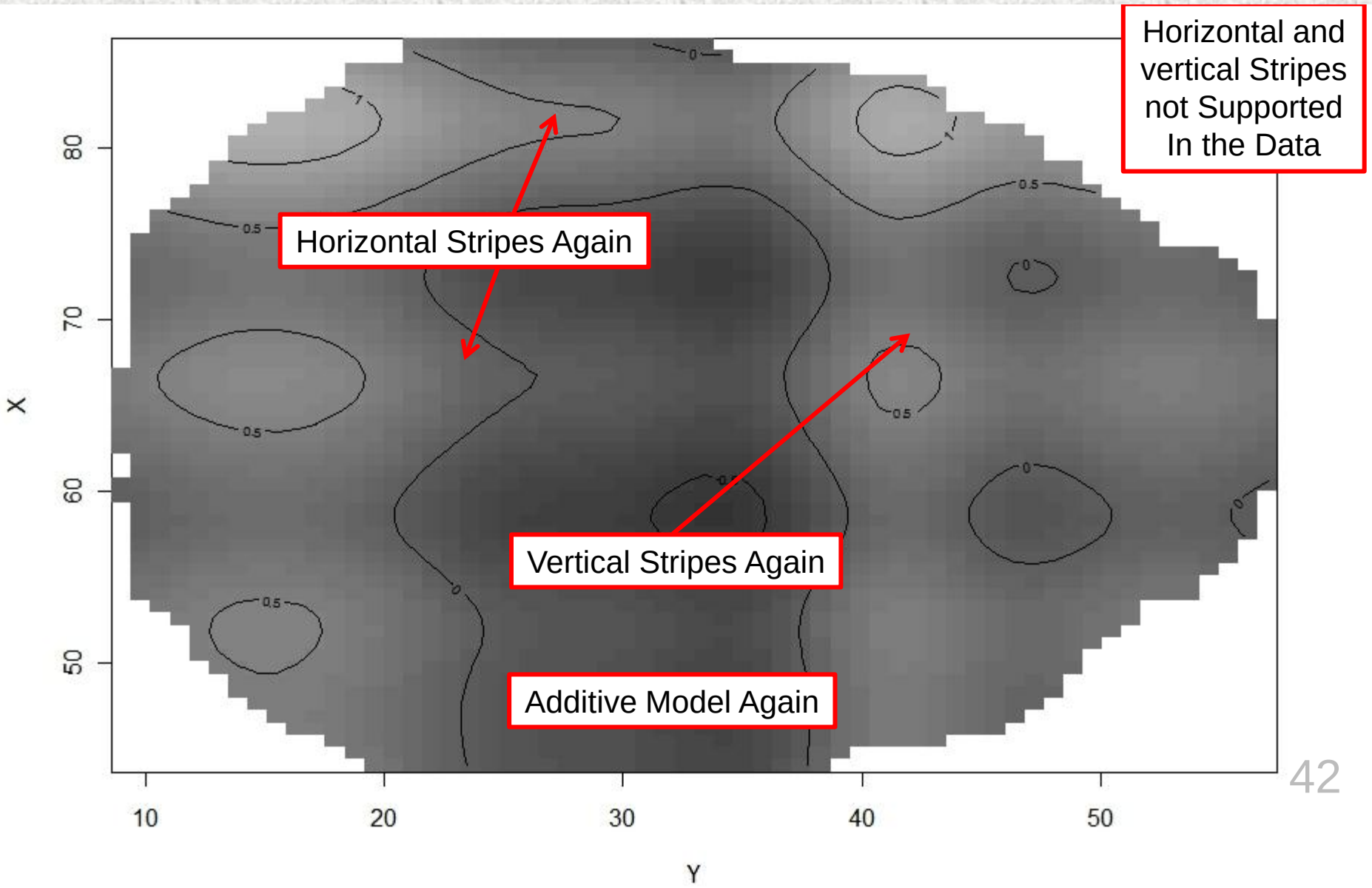
```
> anova(tm1,tm,test="F")

Analysis of Deviance Table

Model 1: medFPQ ~ s(Y, k = 10, bs = "cr") + s(X, bs = "cr", k = 10)
Model 2: medFPQ ~ te(Y, X, k = 10)
  Resid. Df Resid. Dev     Df Deviance      F    Pr(>F)
1    1547.5    1013.17
2    1506.0     914.56 41.471   98.613 4.0015 9.635e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Additive Model Rejected Again

Model `tm` Image Plot with Contours



Additive Model `tm1` Image Plot



Air Pollution in Chicago Example

- Deals with relationship between air pollution and health.
- Air-pollution epidemiology data from Peng and Welty (2004).
  - Data Frame: **`chicago`** in package **`gamair`**

```
> library(mgcv)
> data(chicago)
> chicago
```

Response variable: Daily death rate over a number of years

	death	pm10median	pm25median	o3median	so2median	time	tmpd
1	130	-7.433544304	NA	-19.592337860	1.928042597	-2556.5	31.5
2	150	NA	NA	-19.038613660	-0.985563116	-2555.5	33.0
3	101	-0.826530612	NA	-20.217337860	-1.891416086	-2554.5	33.0
4	135	5.566455696	NA	-19.675671200	6.139341274	-2553.5	29.0
5	126	NA	NA	-19.217337860	2.278464871	-2552.5	32.0
6	130	6.566455696	NA	-17.634004530	9.858583914	-2551.5	40.0

. . .

Note: **`pm10median`** is particulate matter; **`o3median`** are levels of ozone; **`so2median`** levels of sulpher dioxide; **`tmpd`** is mean daily temperature.

Air Pollution in Chicago

- Conventional approach: Assume observed number of deaths are **Poisson** random variables:

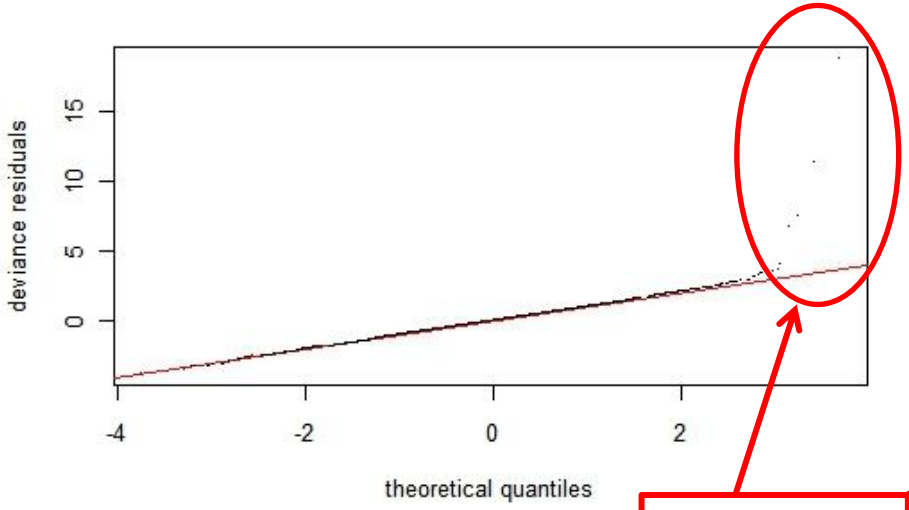
$$\log\left(\mathrm{E}\left[\mathrm{death}_i\right]\right) = f\left(\mathrm{time}_i\right) + \beta_1 \mathrm{pm10median}_i + \beta_2 \mathrm{so2median}_i + \beta_3 \mathrm{o3median}_i + \beta_4 \mathrm{tmpd}_i$$

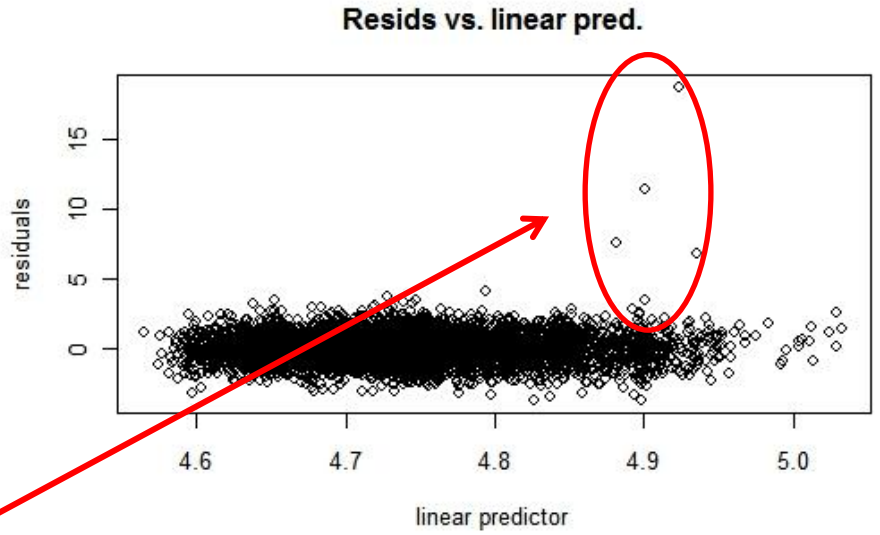
where $\mathbf{death}_i$ follows a Poisson distribution and f is a smooth function.

- We check and fit the model:

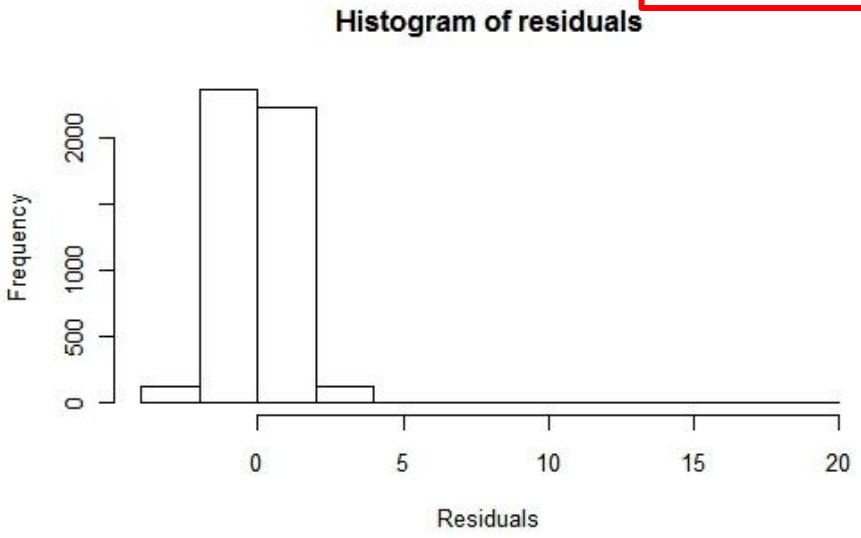
```
> ap0 <- gam(death~s(time,bs="cr",
       k=200)+pm10median+so2median+
       o3median+tmpd,data=chicago,
       family=poisson)

> gam.check(ap0)
```

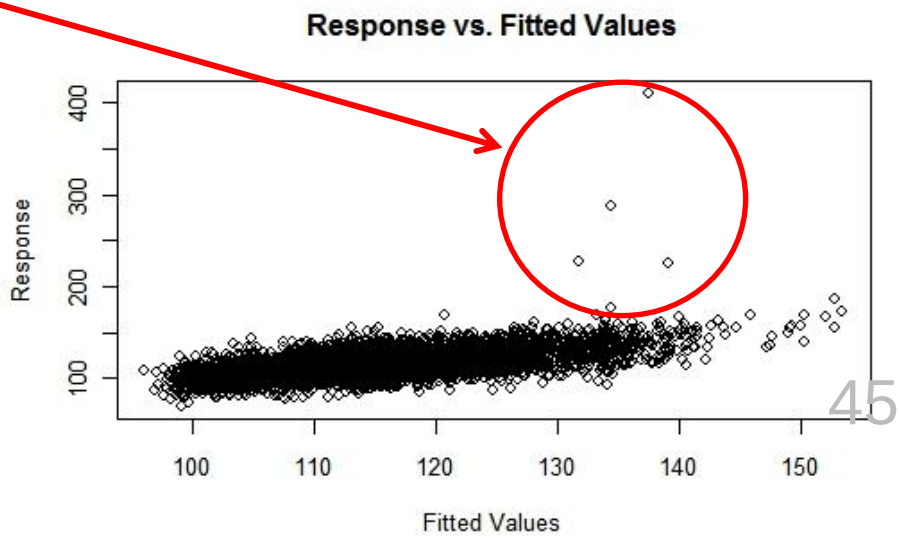
Basic Model Checking Plots for the apo Air Pollution Mortality Model





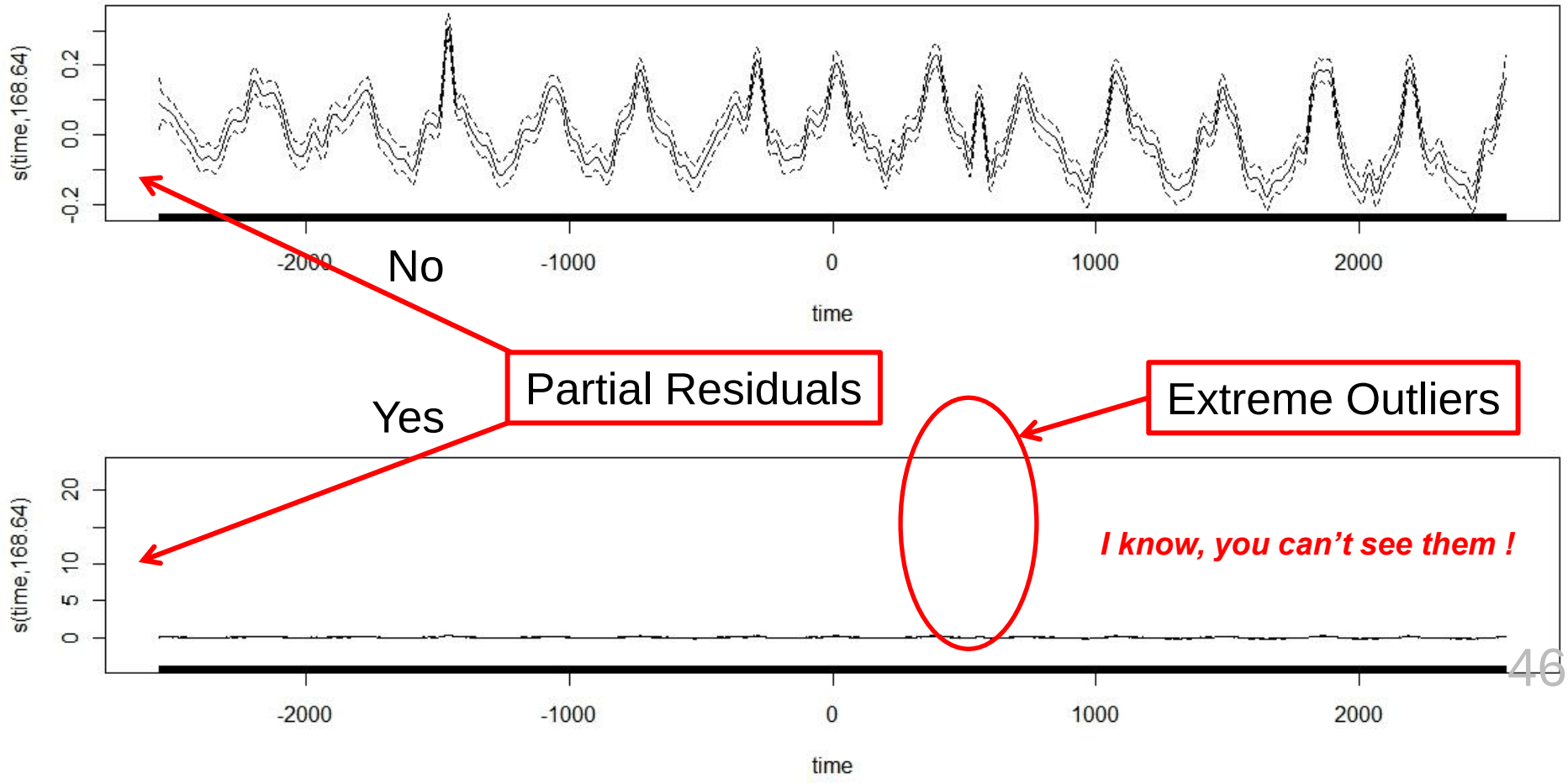
Problems





Estimate of Smooth from Model apo With And Without Residuals

```
> par(mfrow=c(2,1))
> plot(ap0,n=1000)  # n increased to make plot smooth
> plot(ap0,residuals=TRUE,n=1000)
```



Air Pollution in Chicago Model `ap1`

```
> chicago$death[3111:3123]
> [1] 112 97 122 119 116 121 226 411 287 228 159 142 123
```

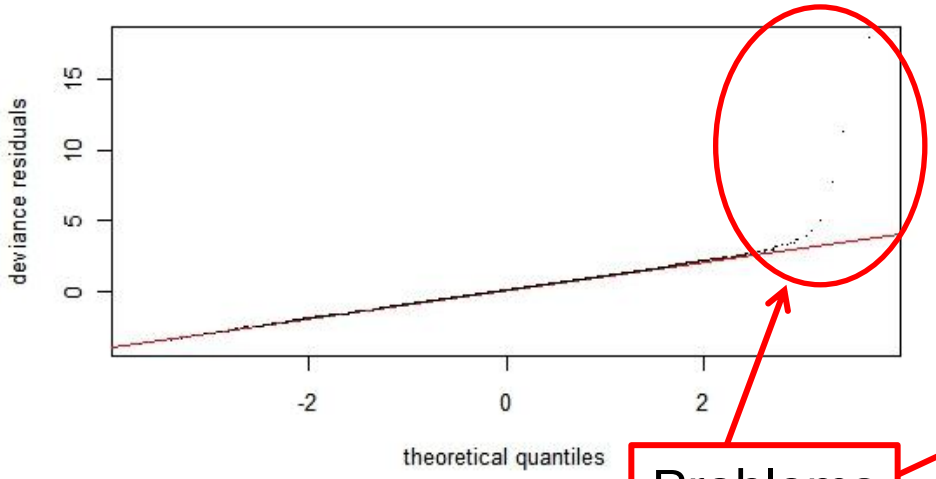
- If we plot this section of data, we see the peak associated with ***very high temperatures*** and ***high ozone***.
  - → Incorporate non-linear effect of temperature and ozone on death rate.
- Model becomes:

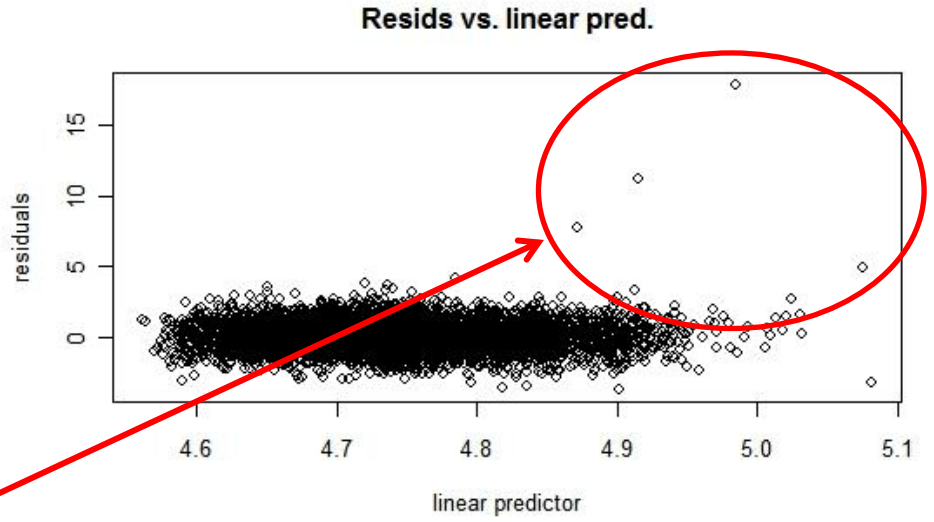
$$\log(\mathrm{E}[\mathrm{death}_i]) = f_1(\mathrm{time}_i) + f_2(\mathrm{pm10median}_i) + f_3(\mathrm{so2median}_i) + f_4(\mathrm{o3median}_i) + f_5(\mathrm{tmpd}_i)$$

where the f_j are smooth functions. We fit the model:

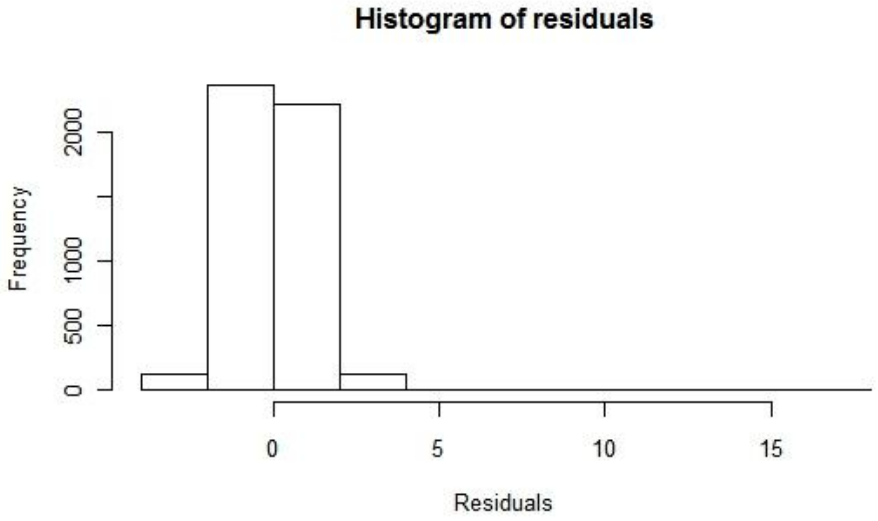
```
> ap1<-gam(death~s(time,bs="cr",k=200)+s(pm10median,bs="cr")+
      s(so2median,bs="cr")+s(o3median,bs="cr")+s(tmpd,bs="cr"),
      data=chicago,family=poisson)
> gam.check(ap1)
```

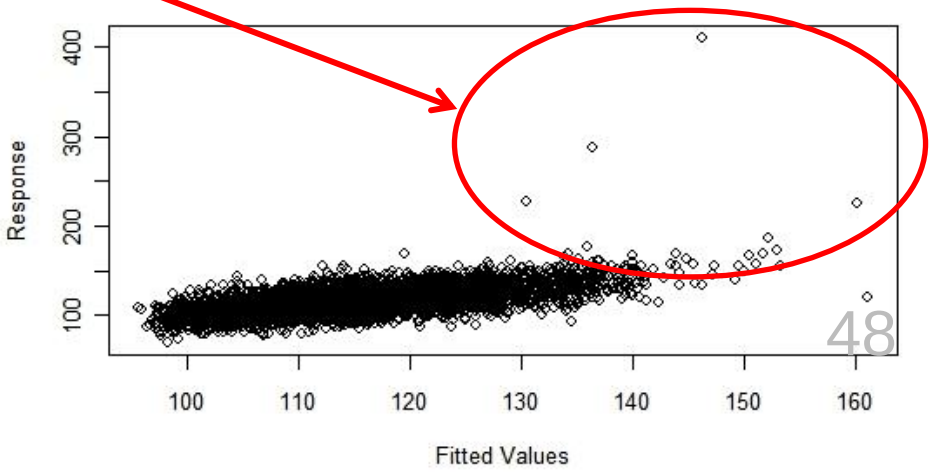
Basic Model Checking Plots for the `ap1` Air Pollution Mortality Model





Problems





Air Pollution in Chicago

- Question becomes: What are the appropriate aggregations and lags of the covariates to predict mortality?
  - Focus on the ***sum*** of pollutant (ozone) and temperature levels over the four days up to and including each daily mortality reading.
- We create this function:

```
lag.sum <- function(a,l0,l1)
## l0 is minimum range of lag, l1 is maximum
{ n<-length(a)
  b<-rep(0,n-l1)
  for (i in 0:(l1-l0)) b <- b + a[(i+1):(n-l1+i)]
  b
}
```

Modified Chicago Air Pollution Data

- We modify the data for death (response) and time of day (predictor):

```
> death<-chicago$death[4:5114]
> time<-chicago$time[4:5114]
```

o3 = ozone;
tmp = temp;
pm10 = part.;
so2 = sul diox;

- We use **lag.sum()** to modify other predictors:

```
> o3 <- lag.sum(chicago$o3median,0,3)
> tmp <- lag.sum(chicago$tmpd,0,3)
> pm10 <- lag.sum(log(chicago$pm10median+40),0,3)
> so2 <- lag.sum(log(chicago$so2median+10),0,3)
```

- Aggregate variables are used to predict death, for example:

$$o3_i = \sum_{j=i-3}^{i} o3median_j$$

Sum of daily ozone for four days

Air Pollution in Chicago Model `ap2`

- Still looking at a combination of high ozone and high temperature (with particulates):

$$\log\left(\mathrm{E}\left[\mathrm{death}_i\right]\right) = f_1\left(\mathrm{time}_i\right) + f_2\left(\mathrm{o3}_i, \mathrm{tmp}_i, \mathrm{pm10}_i\right)$$

3x way

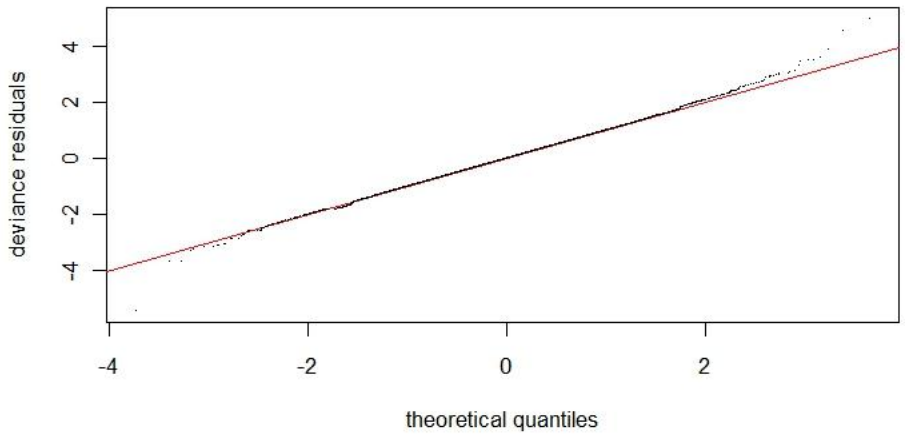
- where f_1 and f_2 are smooth functions. We fit and check the model:

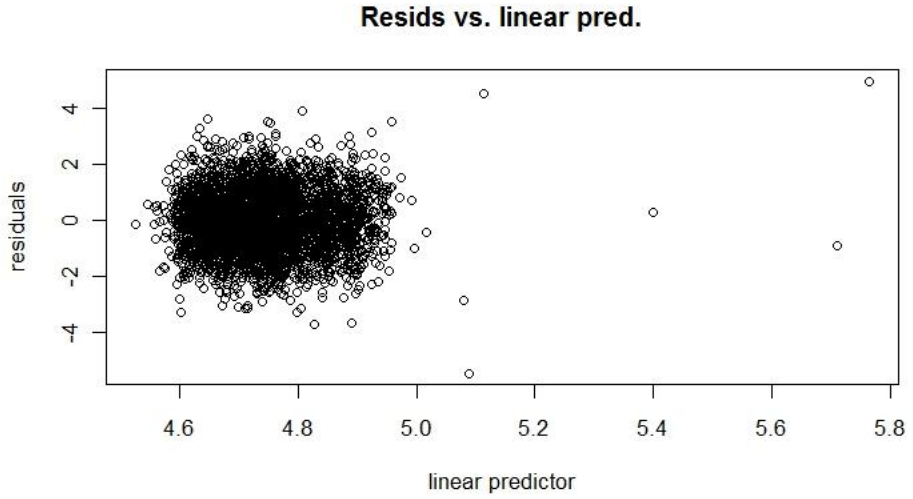
Note: This model is ***very*** slow to converge !

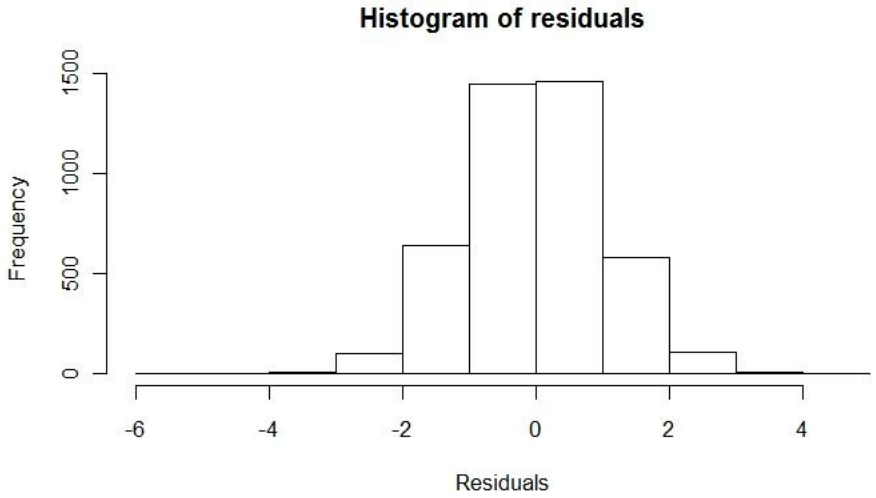
```
> ap2 <- gam(death ~ s(time,bs="cr",k=200) +
                     te(o3,tmp,pm10,k=c(8,8,6)),family=poisson)
> gam.check(ap2)
```

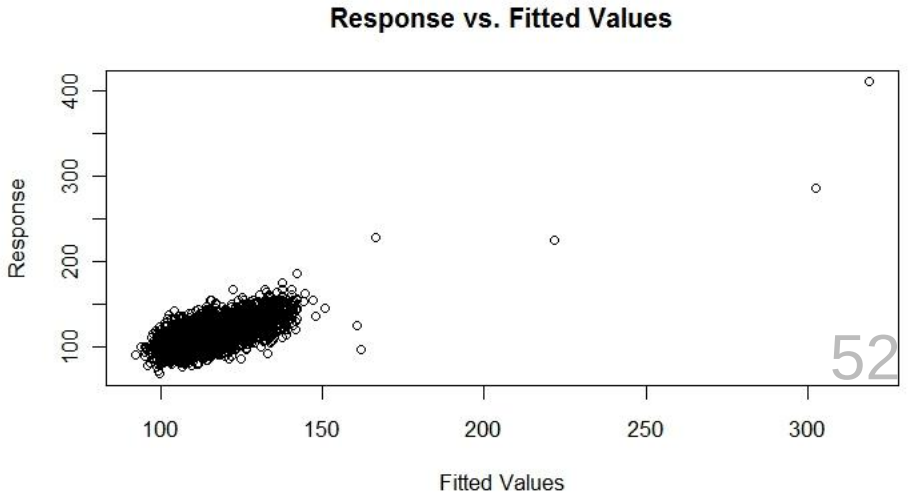
Basic Model Checking Plots for the ap2 Air Pollution Mortality Model

Greatly improved default checking plots









Air Pollution in Chicago Model `ap3`

Note: This model is also slow to converge !

- Simplify **`ap3`** by separating out interaction of ozone and temperature:

$$\log(\mathrm{E}[\mathrm{death}_i]) = f_1(\mathrm{time}_i) + f_2(\mathrm{o3}_i, \mathrm{tmp}_i) + f_3(\mathrm{pm10}_i)$$

2x way

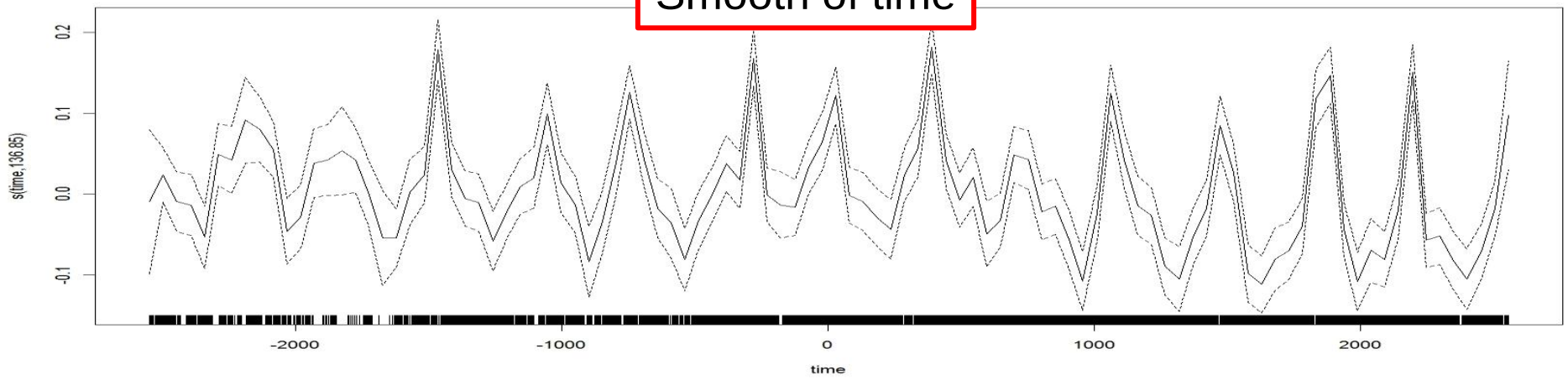
- where f_1, f_2 and f_3 are smooth functions. We fit the model:

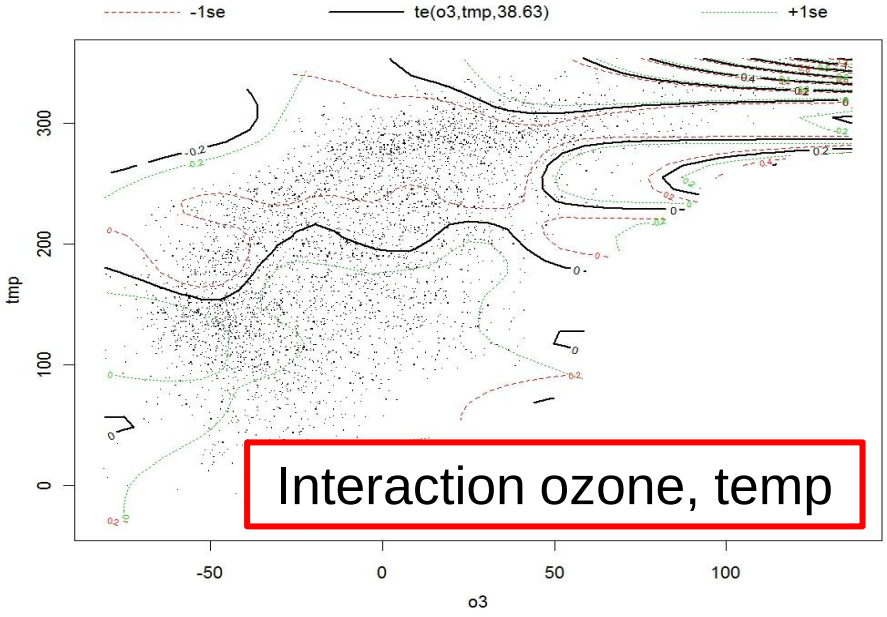
```
> ap3 <- gam(death ~ s(time,bs="cr",k=200) + te(o3,tmp,k=8) +
                     s(pm10,bs="cr",k=6),family=poisson)
```

- By default, smoothness selection uses UBRE
  - **`ap3`** has smallest UBRE
  - Adding **`so2`** very marginally improves the UBRE, but with a very small effect (barely significant).
- We show estimates of components of **`ap3`** on next slide.

Estimates of Components of Chicago Model ap3

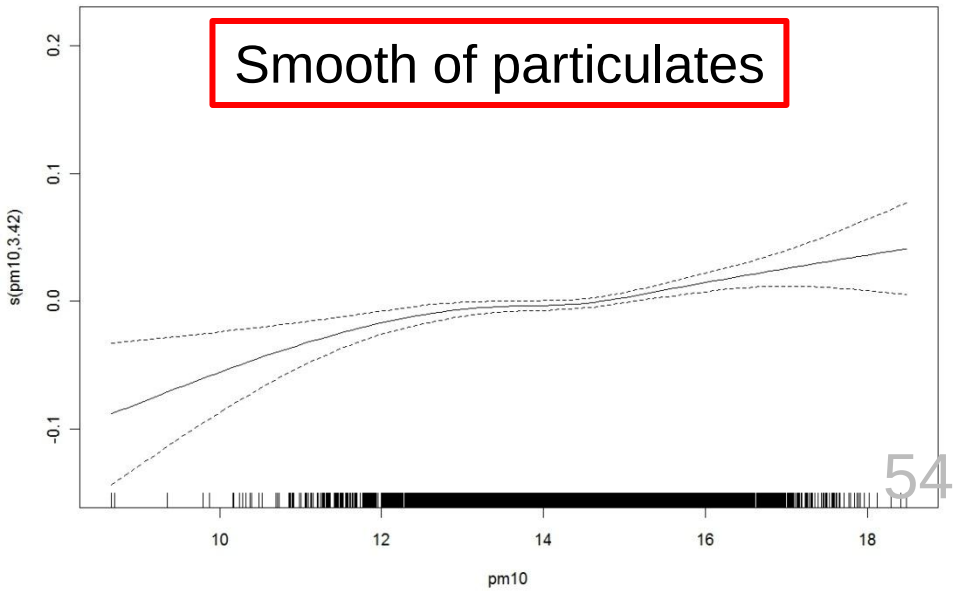
Smooth of time





Interaction ozone, temp

Smooth of particulates



Other GAM Packages

- Are other packages implementing GAMs:
  - Trevor Hastie's **`gam`** package
  - Chong Gu's **`gss`** package
  - **`assist`** package implements Wang & Wang (1998)
  - **`gamlss`** package implements Rigby and Stasinopoulos (2004).
- Package **`gam`**
  - Implements the GAM framework of Hastie and Tibshirani (1990)
  - Package **`mgcv`** (Wood) preceded package **`gam`** and also has functions based closely on the **S** equivalent designed by Hastie (1993).
    - Thus, basic use of **`gam`** function is similar in **`mgcv`** and **`gam`**.
    - Main differences: (1) **`s()`** terms in gam::gam are cubic smoothing spline smooths of one var; (2) Smooths of any number of terms provided by **`lo()`** (loess) smooths; (3) **`gam::gam`** does not estimate degree of smoothness automatically.

Chicago Revisited with `gam` Package

- Fit a version of final Chicago air pollution example:

```
> library(gam)
> install.packages("akima")
> library(akima) # needed for plotting
> bfm <- gam(death~s(time,df=140)+lo(o3,tmp,span=.1),
          family=poisson,control=gam.control(bf.maxit=150))
> summary(bfm)
```

- We provide a default summary of the fit:

```
Call: gam(formula = death ~ s(time, df = 140) + lo(o3, tmp, span = 0.1),
    family = poisson, control = gam.control(bf.maxit = 150))
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-4.05604 -0.72077 -0.02738  0.67355 13.43195

(Dispersion Parameter for poisson family taken to be 1)

    Null Deviance: 9860.715 on 5110 degrees of freedom
Residual Deviance: 5823.157 on 4929.065 degrees of freedom
AIC: 39815.83
```

Continued on next slide . . .

Chicago Revisited with `gam` Package

- Fit a version of final Chicago air pollution example:

```
> bfm <- gam(death~s(time,df=140)+lo(o3,tmp,span=.1),
          family=poisson,control=gam.control(bf.maxit=150))
> summary(bfm)
```

- Continued default summary of the fit:

Continued from previous slide . . .

```
Number of Local Scoring Iterations: 20

DF for Terms and Chi-squares for Nonparametric Effects

                          Df Npar Df Npar Chisq    P(Chi)
(Intercept)                1
s(time, df = 140)          1   139.0    1336.27 < 2.2e-16 ***
lo(o3, tmp, span = 0.1)    2    38.9     645.51 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

gss Package

- Implements General Smoothing Spline Approach of Wahba (1990) and Gu (2002)
  - Somewhat different from other **gam** functions.
  - Based on ANOVA decomposition of functions.
  - Air pollution model again:

```
> install.packages("gss")
> library(gss)
> ssm <- gssanova1(death~time+o3*tmp,family="poisson",
        nbasis=200)
```

Take a long coffee break . . .

Main effects and 2 way

- **gssanova1()** is a "computationally efficient reduced rank version" of function **gssanova()**, which fits generalized smoothing spline ANOVA models (Kim and Gu 2004).
- Model formula specifies a linear predictor with structure:

$$f_i(\text{time}) + f_2(\text{o3}) + f_3(\text{tmp}) + f_4(\text{o3}, \text{tmp})$$